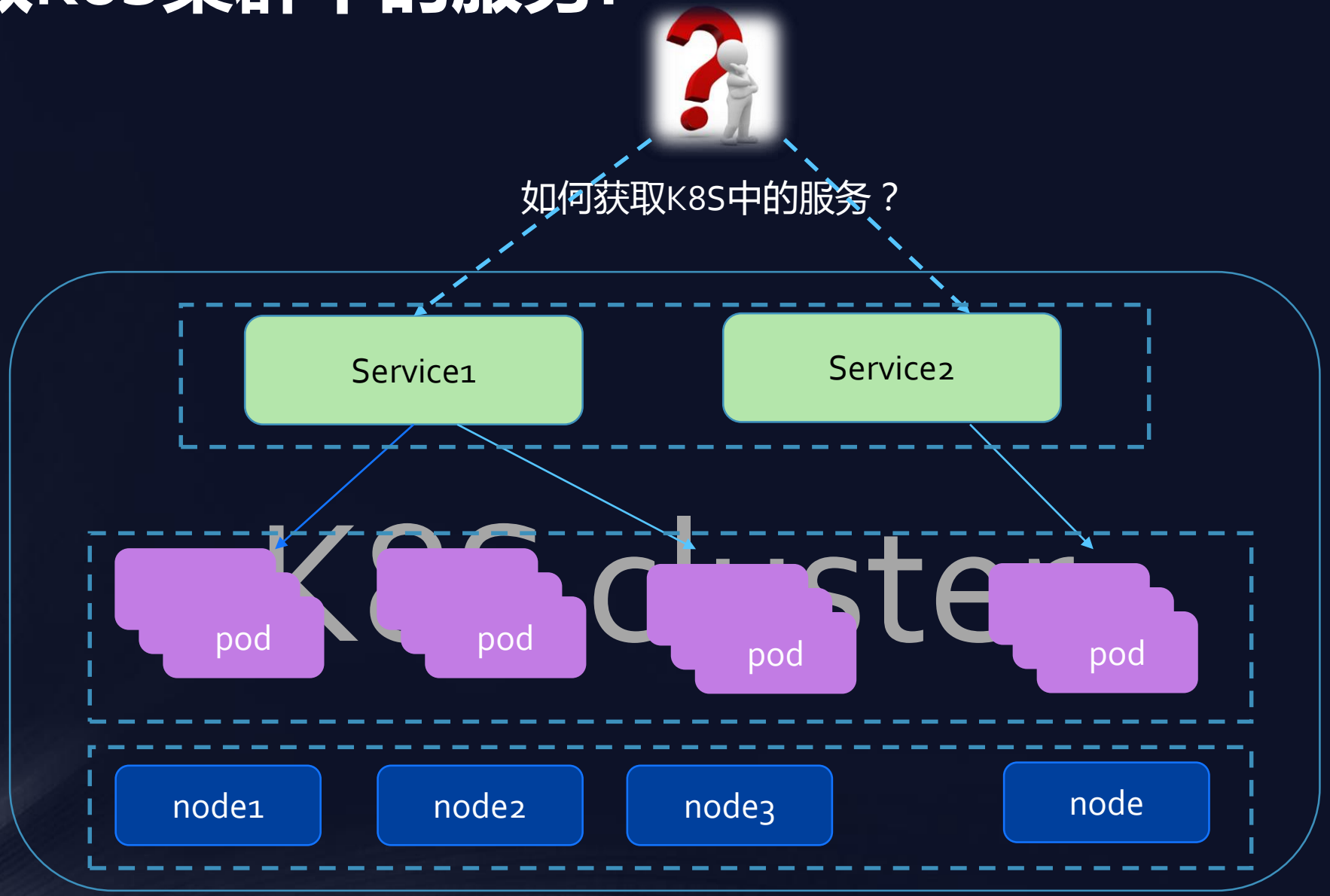
基于Nginx的负载均衡器在k8s中的实践

NOKIA STEVEN OU

Agenda

- K8S集群中为什么需要负载均衡器
- 在K8S集群的边缘节点运行Nginx
- Nginx如何发现K8S中的服务
- K8S中的Ingress

如何获取K8S集群中的服务?



为何获取K8S中的服务是比较困难的？

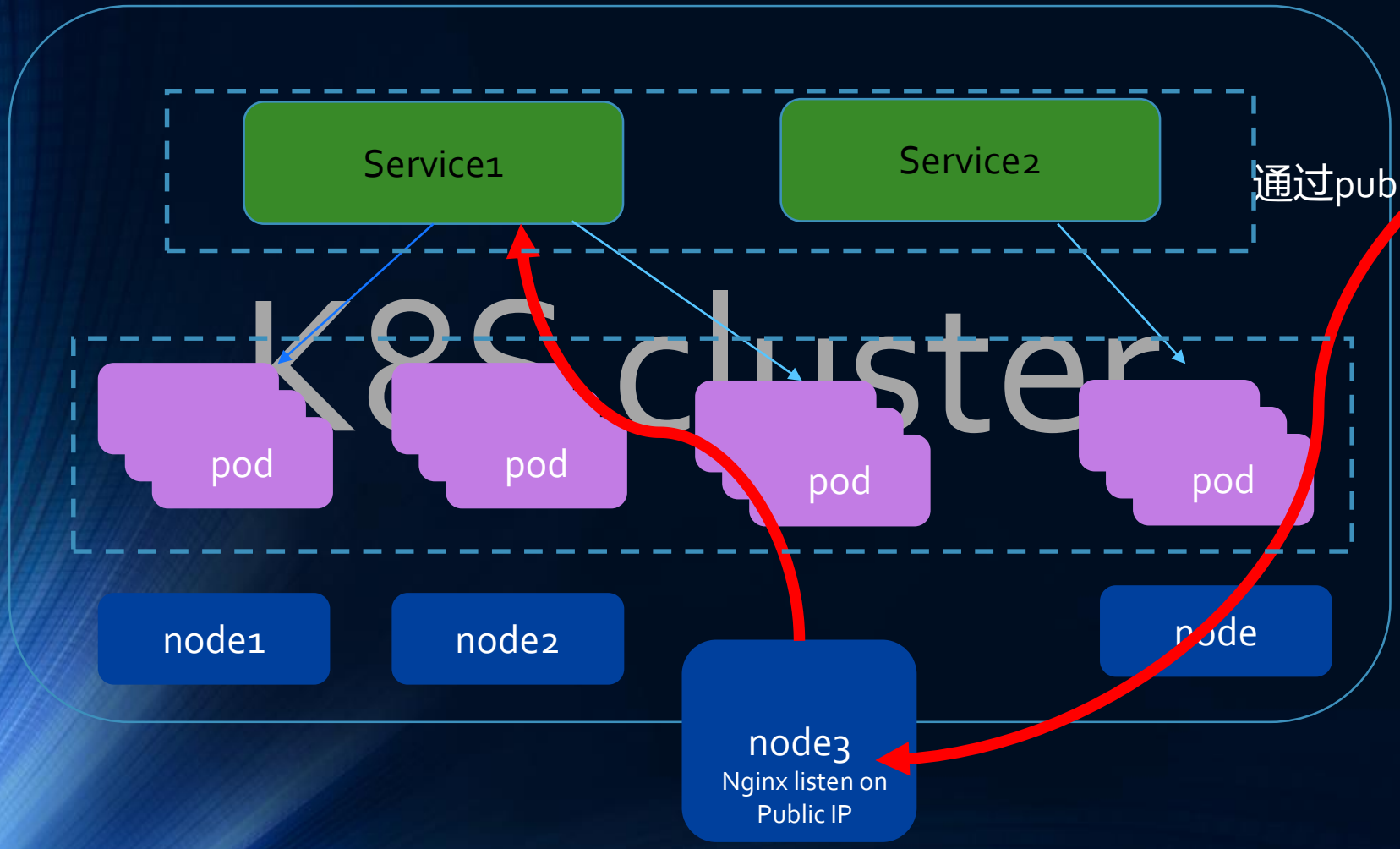
- 每一个pod都有一个由网络层提供的私有地址，在K8S集群中的任一节点上可以可达。K8S集群外部不能直接访问。
- 一组相同功能的pod构成service，K8S赋予每个Service一个cluster IP地址。Service可以从cluster IP地址访问。
- Cluster IP地址只在K8S集群内有效，不能从外部直接访问。

外部应用如何才能访问K8S中的service

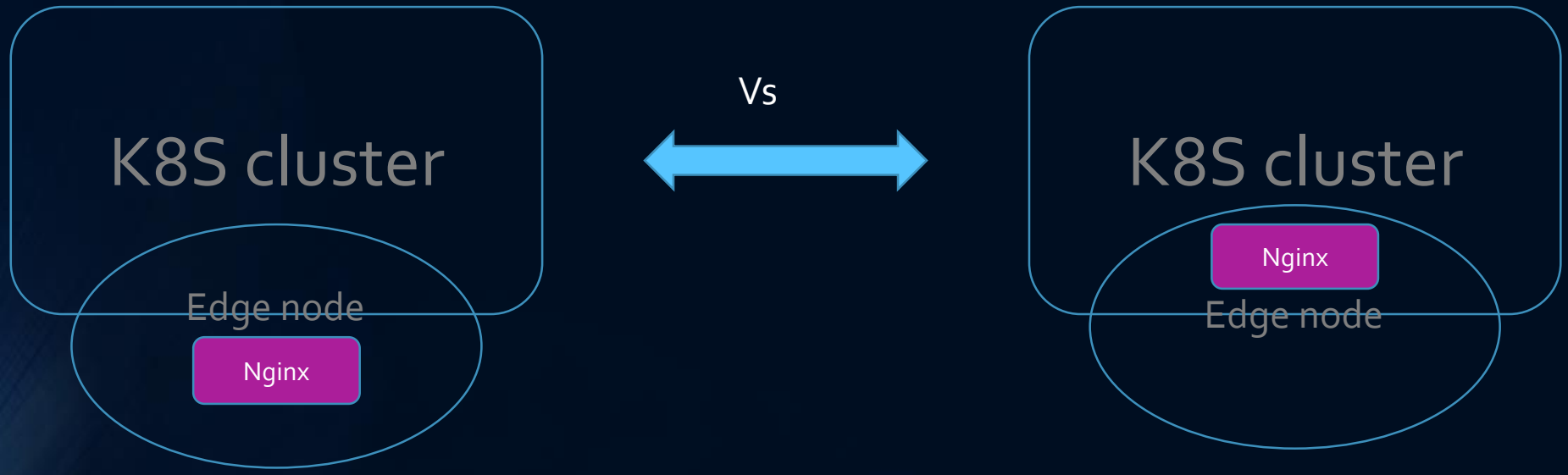
- K8S集群中要有一个或者多个public IP边缘节点
- 外部要访问K8S集群内的Service必须通过边缘节点的public IP地址进行访问
- 有public IP地址的边缘节点需要部署如Nginx，HAProxy等反向代理将请求转发给K8S中的service

如何获取K8S集群中的服务?

通过public IP可以访问K8S中的服务了



如何部署Nginx?



Nginx部署在K8S集群外

Nginx部署在K8S集群内

Nginx部署在K8S集群内和集群外的区别

- Nginx部署在K8S集群内：
  - 在Nginx的配置文件中不用指定nameserver， nameserver已由K8S配置好
  - K8S管理Nginx的启停
  - Nginx监听端口要映射到host的端口上或者使用host网络 (hostNetwork:true)
- Nginx部署在K8S集群外：
  - 需要在Nginx的配置文件中指定nameserver
  - 需要自己管理Nginx的启停
- 推荐：
  - Nginx部署在K8S集群内
  - 以Daemon方式运行

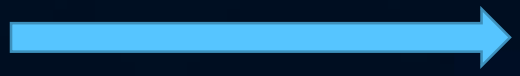
如何让Nginx运行在边缘节点上

- 对边缘节点打标签，表明此节点是一个边缘节点

```
$ kubectl label node node3 node-type=edge
```

- 创建nginx pod时通过nodeSelector将nginx调度到边缘节点上

```
apiVersion: extensions/v1beta1
kind: DaemonSet
metadata:
spec:
 template:
   spec:
     nodeSelector:
      node-type: edge
```



Nginx运行在边缘节点上了

通过Cluster IP 地址发现服务

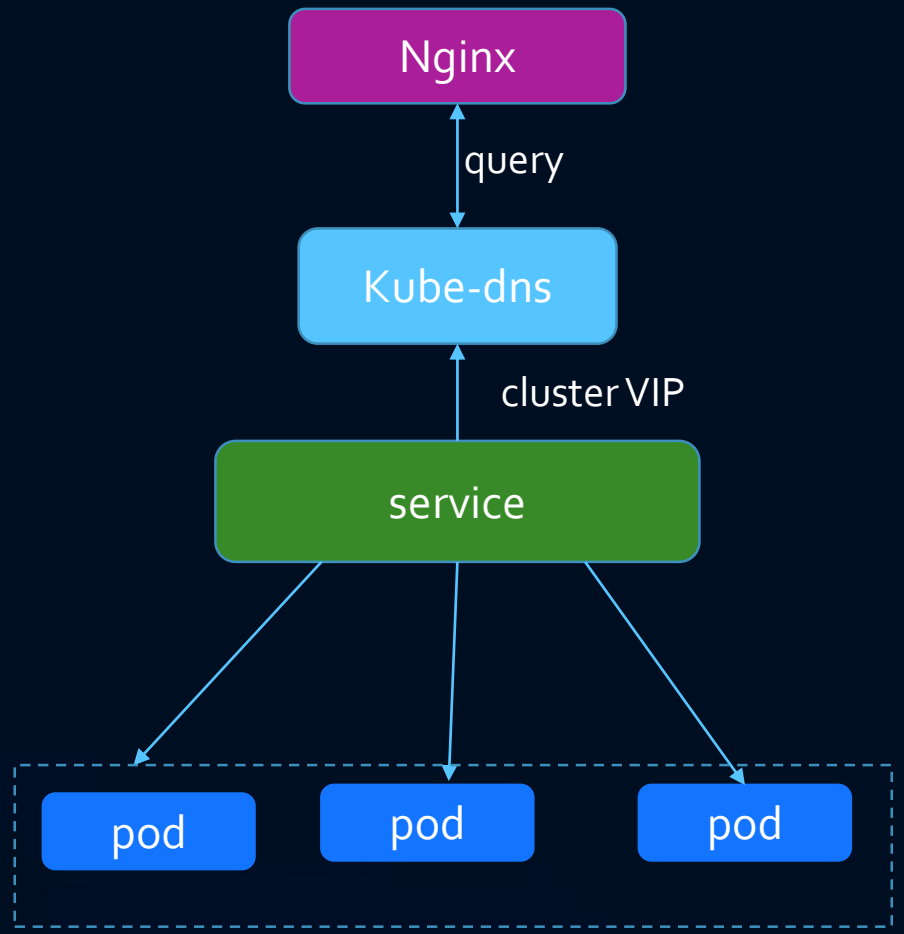
- K8S为服务分配一个cluster VIP, service通过cluster VIP可达
- K8S将服务的名字和cluster VIP放入DNS服务器
- 将服务名配置到nginx.conf

在K8S中发布hello 服务

```
$ kubectl run hello –image=hello –replicas=3 –port=8080
$ kubectl expose deployment hello
```

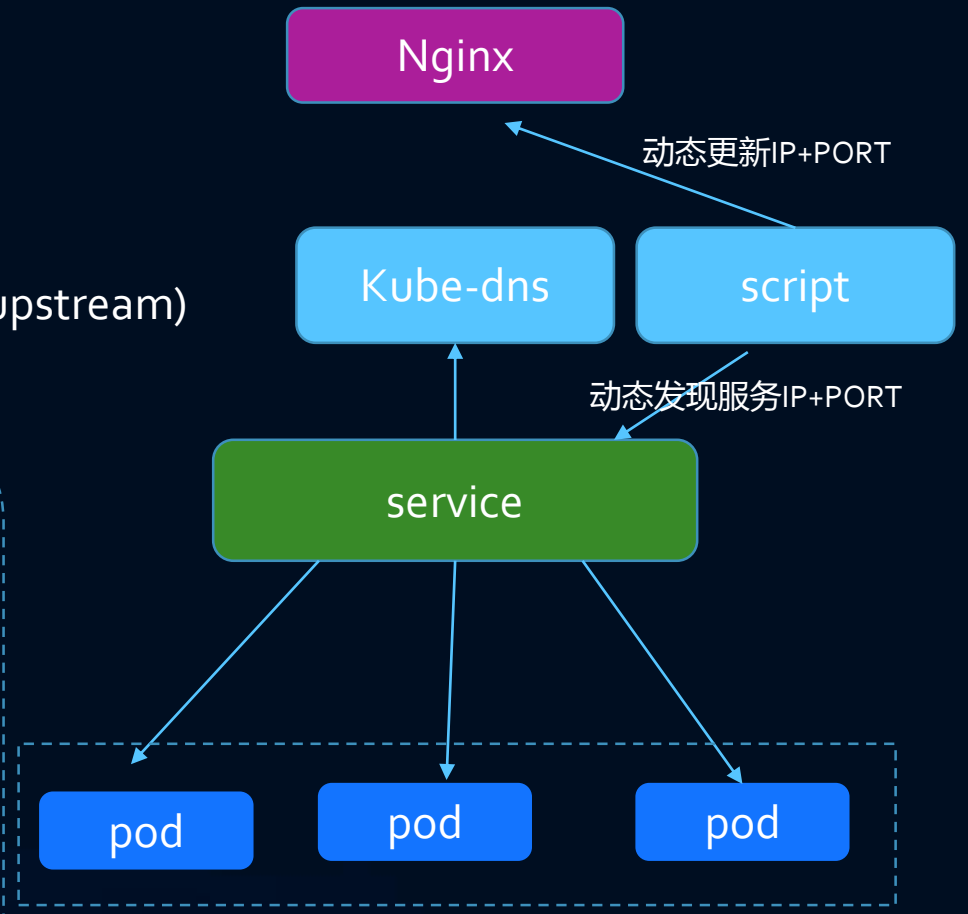
配置nginx:

```
server {
   location / {
         proxy_pass http://hello:8080;
    }
}
```



通过NodePort发现服务

- K8S为服务分配了cluster VIP, service通过cluster VIP可达
- 为service中的POD分配了一个相同的端口，服务通过节点IP+PORT可达
- 可以通过程序发现节点IP地址，动态将地址添加到nginx (ngx_dynamic_upstream)



在K8S中发布hello 服务

```
$ kubectl run hello –image=hello –replicas=3 –port=8080
$ kubectl expose deployment hello –type=NodePort
```

配置nginx:

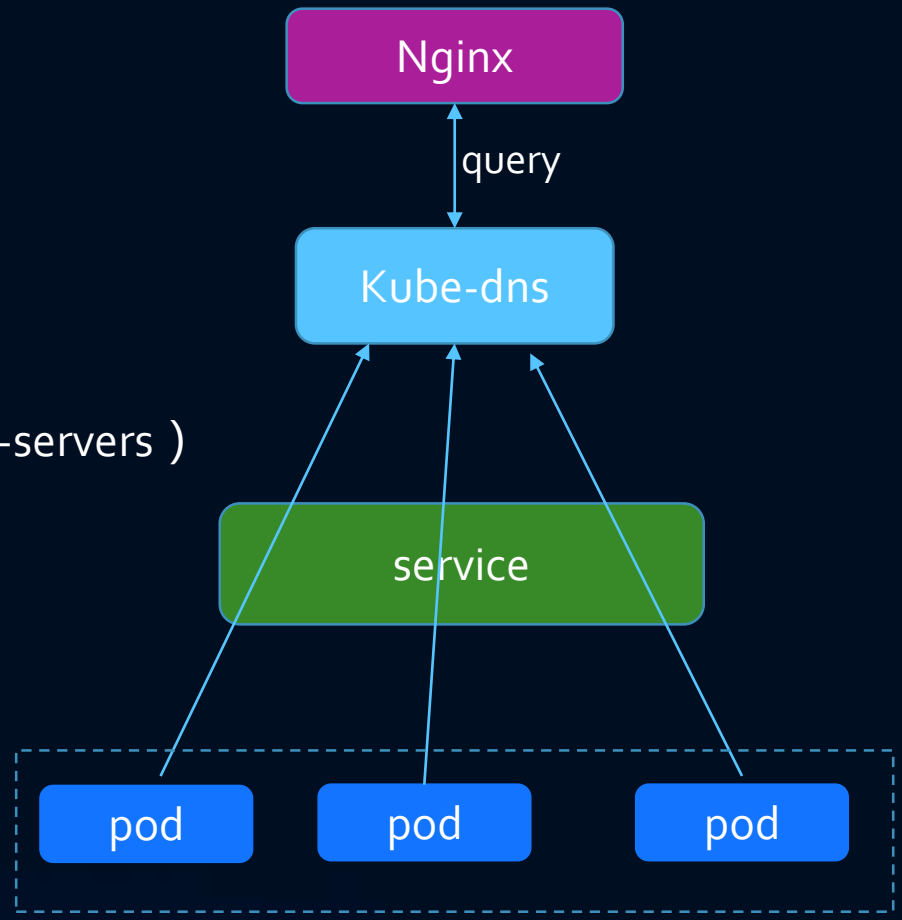
```
upstream my_services {
   zone my_zone 1m;
}
server {
   location / {
          proxy_pass http://my_services;
    }
}
```

Add/remove node to/from nginx:

```
$ curl http://localhost:6000/dynamic?upstream=zone_for_backends&add=&server=10.10.1.1:30900"
$ curl http://localhost:6000/dynamic?upstream=zone_for_backends&remove=&server=10.10.1.1:30900"
```

通过Headless service发现服务(推荐）

- K8S没有为服务分配cluster IP, service无法通过cluster VIP可达
- K8S将服务中的POD IP放入DNS服务器中
- 在DNS中通过服务名查找地址会返回多条IP地址
- Nginx需支持upstream动态后端服务器地址注册(nginx-upstream-dynamic-servers）



在K8S中发布hello 服务

```
$ kubectl run hello –image=hello –replicas=3 –port=8080
$ kubectl expose deployment hello --cluster-ip=None
```

配置nginx:

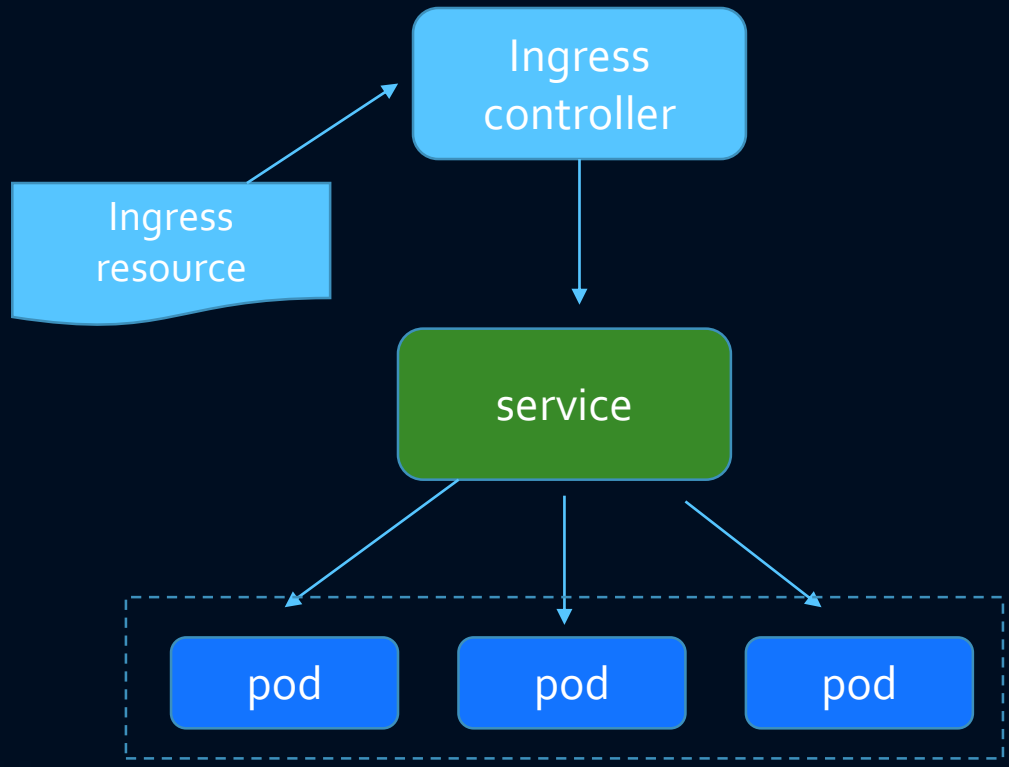
```
upstream my_services {
   server my_services:8080 resolve;
}
server {
   location / {
          proxy_pass http://my_services;
    }
}
```

为什么有 Ingress?

- Loadbalancer是一个很通用的需求，大多数应用都需要Loadbalancer
- 流行的Loadbalancer除了Nginx还有HA proxy
- 不同的Loadbalancer配置格式不一样，启停方式不同...
- 在L7/L4层面提供的功能大同小异
- 能否在L7/L4业务层面提供抽象并适用于不同的Loadbalancer？
- K8S提供了对L7的HTTP模型抽象，用户只需要关注HTTP的业务分发模型，不用考虑不同Loadbalancer的配置，更新差异

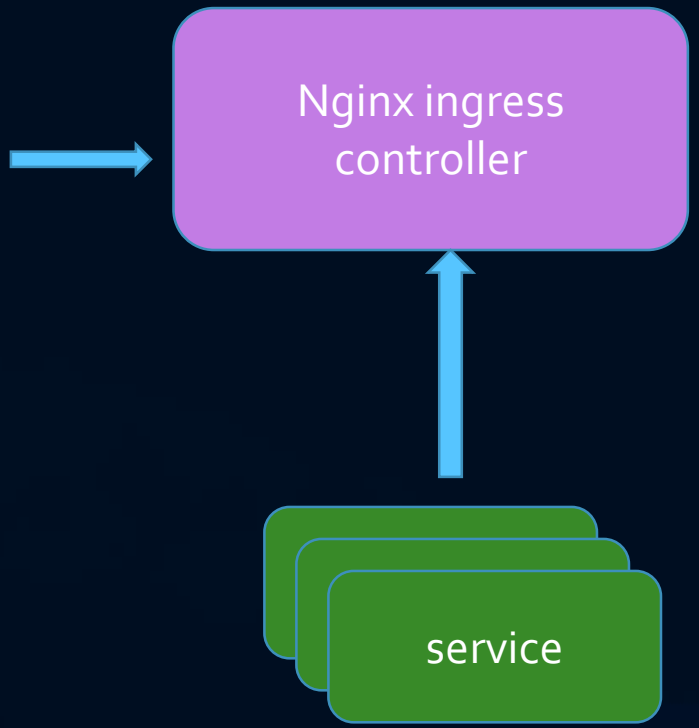
理解Ingress概念

- Ingress 资源定义了如何将HTTP/TCP等请求映射到service
- Ingress控制器读取Ingress资源中定义的请求映射，通过apiserver查找相关的service信息，并更新Loadbalancer配置
- 用户通过获取Ingress控制器的public IP就可以访问K8S中服务了
- 现有的Ingress控制器：nginx, haproxy, GCE, traefik



Ingress 资源定义文件

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: test
spec:
  rules:
  - host: foo.bar.com
    http:
      paths:
      - path: /foo
        backend:
          serviceName: s1
          servicePort: 80
      - path: /bar
        backend:
          serviceName: s2
          servicePort: 80
```




```
upstream s1_backends {
    server s1:80 resolve;
}
upstream s2_backends {
    server s2:52 resolve;
}

server  {
   listen 8080;
   location /foo {
        proxy_pass http://s1_backends;
   }
  location /bar {
        proxy_pass http://s2_backends;
   }
}
```

Q & A