主讲老师：Fox

> **有道笔记地址：**https://note.youdao.com/s/3aBJYuGf

用k8s部署电商项目微服务

tulingmall-product

以**product服务**为例，我们来创建对应的**deployment和service**，做之前需要把商品服务做成镜像推到docker镜像仓库里去，参考docker 课程推送镜像到阿里云镜像仓库。

```
$ docker login --username=fox666 registry.cn-hangzhou.aliyuncs.com
$ docker tag [ImageId] registry.cn-hangzhou.aliyuncs.com/fox666/tulingmall-product:[镜像版本号]
$ docker push registry.cn-hangzhou.aliyuncs.com/fox666/tulingmall-product:[镜像版本号]
```

新增文件tulingmall-product-deployment.yaml，内容如下：

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: tulingmall-product-deployment
  labels:
    app: tulingmall-product
spec:
  replicas: 2
  selector:
    matchLabels:
      app: tulingmall-product
  template:
    metadata:
      labels:
        app: tulingmall-product
    spec:
      hostNetwork: true  # 主机模式，初学者建议使用此模式
```

```
      imagePullSecrets:
      - name: myregistrykey  #私有仓库的secret
      containers:
        - name: tulingmall-product
          image: registry.cn-hangzhou.aliyuncs.com/fox666/tulingmall-product:0.0.5
          imagePullPolicy: Always
          ports:
            - containerPort: 8866
          env:
            - name: TZ
              value: Asia/Shanghai
            - name: spring.cloud.nacos.config.server-addr
              value: 192.168.65.174:8848
            - name: LOG_FILE
              value: /var/logs
          volumeMounts:
            - mountPath: /var/logs
              name: log-volume
      volumes:
        - name: log-volume
          hostPath:
            path: /mydata/k8s-app/tulingmall-product/logs
```

> imagePullPolicy可以使用以下3种策略值:
> Always: 默认值，每次创建pod都会重新拉取一次镜像;
> IfNotPresent: 镜像在宿主机上不存在时才拉取;
> Never: 永远不会主动拉取镜像，使用本地镜像，需要你手动拉取镜像下来;

执行如下命令创建商品服务的deployment:

```
kubectl apply -f tulingmall-product-deployment.yaml

#查看部署pod的详细信息
 kubectl describe pod xxxx
```

新增文件tulingmall-product-service.yaml，内容如下:

```yaml
apiVersion: v1
kind: Service
metadata:
  name: tulingmall-product-service
spec:
  type: NodePort
  selector:
    app: tulingmall-product
  ports:
  - name: http
    protocol: TCP
    port: 8866
    targetPort: 8866
```

执行如下命令创建商品服务的service：

```
kubectl apply -f tulingmall-product-service.yaml

#查看部署service的详细信息
 kubectl describe svc xxxx
```

查看商品服务的对外暴露端口：

访问下查询商品的接口，如果有json数据返回，代表服务正常：
http://192.168.65.180:30997/pms/productInfo/27

```
http://192.168.65.210:31357/pms/productInfo/1
```

用相同的方法部署下order，member，gateway，authcenter等服务，这里不一一详述了，附上每个服务k8s部署的yaml文件供大家参考。

tulingmall-authcenter

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: tulingmall-authcenter-deployment
  labels:
    app: tulingmall-authcenter
spec:
  replicas: 1
  selector:
    matchLabels:
      app: tulingmall-authcenter
  template:
    metadata:
      labels:
        app: tulingmall-authcenter
    spec:
      hostNetwork: true
      imagePullSecrets:
      - name: myregistrykey
      containers:
        - name: tulingmall-authcenter
          image: registry.cn-hangzhou.aliyuncs.com/fox666/tulingmall-authcenter:0.0.5
          imagePullPolicy: Always
          ports:
            - containerPort: 9999
          env:
            - name: TZ
              value: Asia/Shanghai
            - name: spring.cloud.nacos.config.server-addr
              value: 192.168.65.174:8848
            - name: LOG_FILE
              value: /var/logs
          volumeMounts:
            - mountPath: /var/logs
              name: log-volume
      volumes:
        - name: log-volume
          hostPath:
```

```
            path: /mydata/k8s-app/tulingmall-authcenter/logs
            type: DirectoryOrCreate

---
apiVersion: v1
kind: Service
metadata:
  name: tulingmall-authcenter-service
spec:
  type: NodePort
  selector:
    app: tulingmall-authcenter
  ports:
  - name: http
    protocol: TCP
    port: 9999
    targetPort: 9999

```

tulingmall-gateway

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: tulingmall-gateway-deployment
  labels:
    app: tulingmall-gateway
spec:
  replicas: 1
  selector:
    matchLabels:
      app: tulingmall-gateway
  template:
```

```yaml
    metadata:
      labels:
        app: tulingmall-gateway
    spec:
      hostNetwork: true
      imagePullSecrets:
      - name: myregistrykey
      containers:
        - name: tulingmall-gateway
          image: registry.cn-hangzhou.aliyuncs.com/fox666/tulingmall-gateway:0.0.5
          imagePullPolicy: Always
          ports:
            - containerPort: 8888
          env:
            - name: TZ
              value: Asia/Shanghai
            - name: spring.cloud.nacos.config.server-addr
              value: 192.168.65.174:8848
            - name: LOG_FILE
              value: /var/logs
          volumeMounts:
            - mountPath: /var/logs
              name: log-volume
      volumes:
        - name: log-volume
          hostPath:
            path: /mydata/k8s-app/tulingmall-gateway/logs
            type: DirectoryOrCreate

---
apiVersion: v1
kind: Service
metadata:
  name: tulingmall-gateway-service
spec:
  type: NodePort
  selector:
    app: tulingmall-gateway
  ports:
  - name: http
```

```
    protocol: TCP
    port: 8888
    targetPort: 8888

```

tulingmall-order-curr

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: tulingmall-order-curr-deployment
  labels:
    app: tulingmall-order-curr
spec:
  replicas: 1
  selector:
    matchLabels:
      app: tulingmall-order-curr
  template:
    metadata:
      labels:
        app: tulingmall-order-curr
    spec:
      hostNetwork: true
      imagePullSecrets:
      - name: myregistrykey
      containers:
        - name: tulingmall-order-curr
          image: registry.cn-hangzhou.aliyuncs.com/fox666/tulingmall-order-curr:0.0.5
          imagePullPolicy: Always
          ports:
            - containerPort: 8844
          env:
            - name: TZ
              value: Asia/Shanghai
            - name: spring.cloud.nacos.config.server-addr
```

```
              value: 192.168.65.174:8848
            - name: LOG_FILE
              value: /var/logs
          volumeMounts:
            - mountPath: /var/logs
              name: log-volume
      volumes:
        - name: log-volume
          hostPath:
            path: /mydata/k8s-app/tulingmall-order-curr/logs
            type: DirectoryOrCreate

---

apiVersion: v1
kind: Service
metadata:
  name: tulingmall-order-curr-service
spec:
  type: NodePort
  selector:
    app: tulingmall-order-curr
  ports:
  - name: http
    protocol: TCP
    port: 8844
    targetPort: 8844
```

tulingmall-cart

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: tulingmall-cart-deployment
  labels:
```

```yaml
    app: tulingmall-cart
spec:
  replicas: 1
  selector:
    matchLabels:
      app: tulingmall-cart
  template:
    metadata:
      labels:
        app: tulingmall-cart
    spec:
      hostNetwork: true
      imagePullSecrets:
      - name: myregistrykey
      containers:
        - name: tulingmall-cart
          image: registry.cn-hangzhou.aliyuncs.com/fox666/tulingmall-cart:0.0.5
          imagePullPolicy: Always
          ports:
            - containerPort: 8855
          env:
            - name: TZ
              value: Asia/Shanghai
            - name: spring.cloud.nacos.config.server-addr
              value: 192.168.65.174:8848
            - name: LOG_FILE
              value: /var/logs
          volumeMounts:
            - mountPath: /var/logs
              name: log-volume
      volumes:
        - name: log-volume
          hostPath:
            path: /mydata/k8s-app/tulingmall-cart/logs
            type: DirectoryOrCreate

---
apiVersion: v1
kind: Service
metadata:
```

```
  name: tulingmall-cart-service
spec:
  type: NodePort
  selector:
    app: tulingmall-cart
  ports:
  - name: http
    protocol: TCP
    port: 8855
    targetPort: 8855

```

tulingmall-unqid

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: tulingmall-unqid-deployment
  labels:
    app: tulingmall-unqid
spec:
  replicas: 1
  selector:
    matchLabels:
      app: tulingmall-unqid
  template:
    metadata:
      labels:
        app: tulingmall-unqid
    spec:
      hostNetwork: true
      imagePullSecrets:
      - name: myregistrykey
      containers:
        - name: tulingmall-unqid
          image: registry.cn-hangzhou.aliyuncs.com/fox666/tulingmall-unqid:0.0.5
```

```
          imagePullPolicy: Always
          ports:
            - containerPort: 8833
          env:
            - name: TZ
              value: Asia/Shanghai
            - name: spring.cloud.nacos.config.server-addr
              value: 192.168.65.174:8848
            - name: LOG_FILE
              value: /var/logs
          volumeMounts:
            - mountPath: /var/logs
              name: log-volume
      volumes:
        - name: log-volume
          hostPath:
            path: /mydata/k8s-app/tulingmall-unqid/logs
            type: DirectoryOrCreate

---
apiVersion: v1
kind: Service
metadata:
  name: tulingmall-unqid-service
spec:
  type: NodePort
  selector:
    app: tulingmall-unqid
  ports:
  - name: http
    protocol: TCP
    port: 8833
    targetPort: 8833

```

tulingmall-member

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: tulingmall-member-deployment
  labels:
    app: tulingmall-member
spec:
  replicas: 1
  selector:
    matchLabels:
      app: tulingmall-member
  template:
    metadata:
      labels:
        app: tulingmall-member
    spec:
      hostNetwork: true
      imagePullSecrets:
      - name: myregistrykey
      containers:
        - name: tulingmall-member
          image: registry.cn-hangzhou.aliyuncs.com/fox666/tulingmall-member:0.0.5
          imagePullPolicy: Always
          ports:
            - containerPort: 8877
          env:
            - name: TZ
              value: Asia/Shanghai
            - name: spring.cloud.nacos.config.server-addr
              value: 192.168.65.174:8848
            - name: LOG_FILE
              value: /var/logs
          volumeMounts:
            - mountPath: /var/logs
              name: log-volume
      volumes:
        - name: log-volume
          hostPath:
            path: /mydata/k8s-app/tulingmall-member/logs
```

```
            type: DirectoryOrCreate

---
apiVersion: v1
kind: Service
metadata:
  name: tulingmall-member-service
spec:
  type: NodePort
  selector:
    app: tulingmall-member
  ports:
  - name: http
    protocol: TCP
    port: 8877
    targetPort: 8877

```

创建Ingress网关服务

Ingress 是整个 K8S 集群的接入层，负责集群内外通讯。

Ingress安装：https://kubernetes.github.io/ingress-nginx/deploy/#quick-start
查看安装是否成功

```
kubectl get pods -n ingress-nginx -owide
```

最后，我们来创建网关服务的Ingress，创建文件tulingmall-gateway-ingress.yaml，内容如下：

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: tulingmall-gateway-ingress
spec:
```

```
  rules:
  - host: gateway.tuling.com
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: tulingmall-gateway-service
            port:
              number: 8888
  ingressClassName: nginx
```

执行如下命令生效规则：

```
kubectl apply -f tulingmall-gateway-ingress.yaml
```

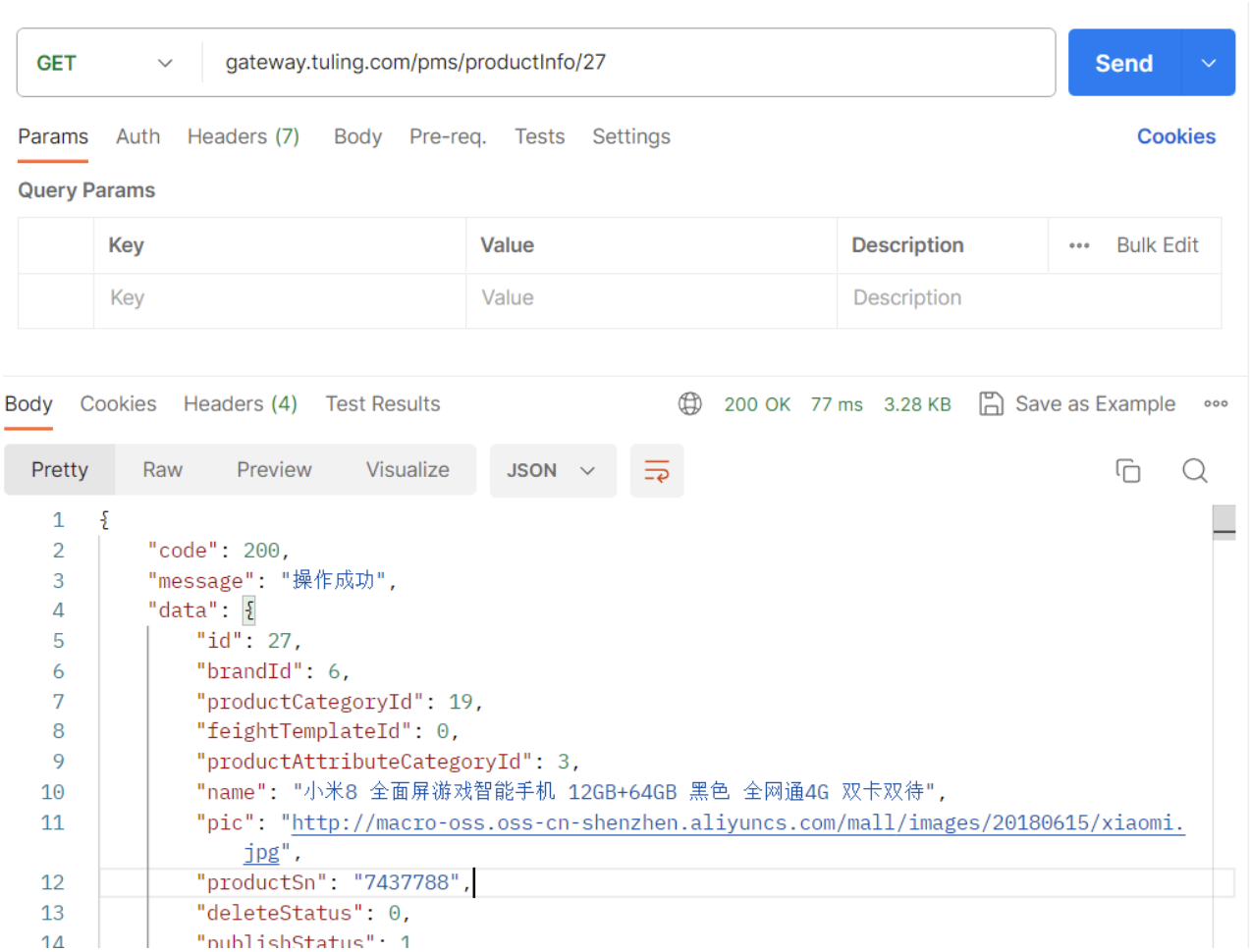
查看生效的ingress规则：

```
kubectl get ing
```

在访问机器配置host，win10客户机在目录：C:\Windows\System32\drivers\etc，在host里增加如下host(ingress部署的机器ip对应访问的域名)

```
192.168.65.137  gateway.tuling.com
```

配置完后直接在客户机用域名请求下网关：



利用EndPoint访问k8s集群外部中间件

K8S中如何使用外部有状态的服务，比如：MySQL、Nacos等。集群内部访问外部数据库或者中间件一般采用endpoints与service关联方式映射。

验证k8s DNS是否可用

```
# kubectl run curl --image=radial/busyboxplus:curl -it
```

进入后执行nslookup kubernetes.default确认解析正常:

```
[ root@curl:/ ]$ nslookup kubernetes.default
Server:    10.96.0.10
Address 1: 10.96.0.10 kube-dns.kube-system.svc.cluster.local

Name:      kubernetes.default
```

```
Address 1: 10.96.0.1 kubernetes.default.svc.cluster.local

```

mysql

以mysql为例手动创建service及endpoint，引入外部mysql，然后通过k8s集群中的域名解析服务访问，访问的主机名格式为：[svc_name].[namespace_name].svc.cluster.local，简写 [svc_name].[namespace_name]

```yaml
apiVersion: v1
kind: Service
metadata:
  name: mysql-product
spec:
  ports:
  - name: mysql-product
    port: 3306
    protocol: TCP
    targetPort: 3306
  type: NodePort

---
apiVersion: v1
kind: Endpoints
metadata:
  name: mysql-product
subsets:
- addresses:
  - ip: 192.168.65.71
  ports:
  - name: mysql-product
    port: 3306
    protocol: TCP
```

执行如下命令

```
kubectl apply -f mysql-product.yaml
# 如果指定了命名空间，比如tulingmall
kubectl apply -f mysql-product.yaml -n tulingmall
```

测试：容器内部可以正常访问mysql-product.tulingmall

nacos

```
apiVersion: v1
kind: Service
metadata:
  name: nacos
spec:
  ports:
    - port: 8848
      name: nacos
      targetPort: 8848
    - port: 9848
      name: client-rpc
      targetPort: 9848
    - port: 9849
      name: raft-rpc
      targetPort: 9849
  type: NodePort

---
apiVersion: v1
kind: Endpoints
metadata:
  name: nacos
subsets:
- addresses:
  - ip: 192.168.65.174
  ports:
   - port: 8848
     name: nacos
```

```
    - port: 9848
      name: client-rpc
    - port: 9849
      name: raft-rpc
```

tulingmall-product

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: tulingmall-product-deployment
  labels:
    app: tulingmall-product
spec:
  replicas: 2
  selector:
    matchLabels:
      app: tulingmall-product
  template:
    metadata:
      labels:
        app: tulingmall-product
    spec:
      #hostNetwork: true
      imagePullSecrets:
      - name: myregistrykey
      containers:
        - name: tulingmall-product
          image: registry.cn-hangzhou.aliyuncs.com/fox666/tulingmall-product:0.0.5
          imagePullPolicy: Always
          ports:
            - containerPort: 8866
          env:
            - name: TZ
              value: Asia/Shanghai
            - name: spring.cloud.nacos.config.server-addr
              value: nacos.default:8848
```

```
            - name: LOG_FILE
              value: /var/logs
          volumeMounts:
            - mountPath: /var/logs
              name: log-volume
      volumes:
        - name: log-volume
          hostPath:
            path: /mydata/k8s-app/tulingmall-product/logs

---

apiVersion: v1
kind: Service
metadata:
  name: tulingmall-product-service
spec:
  type: NodePort
  selector:
    app: tulingmall-product
  ports:
  - name: http
    protocol: TCP
    port: 8866
    targetPort: 8866
```

常见错误

1. 无法从私有镜像仓库拉取镜像，抛出如下错误：

Failed to pull image "registry.cn-hangzhou.aliyuncs.com/fox666/tulingmall-member:0.0.5": rpc error: code = Unknown desc = Error response from daemon: pull access denied for registry.cn-hangzhou.aliyuncs.com/fox666/tulingmall-member, repository does not exist or may require 'docker login': denied: requested access to the resource is denied

解决方案：使用 docker 的用户信息来生成 secret：

```
kubectl create secret docker-registry myregistrykey --docker-server=registry.cn-hangzhou.aliyuncs.com --docker-username=fox666 --docker-password=xxx
```

> 参数含义：
> myregistrykey: 指定密钥的键名称, 自定义
> docker-server: 指定 docker 仓库地址
> docker-username: 指定 docker 仓库账号
> docker-password: 指定 docker 仓库密码

在创建 Pod 的时候，通过imagePullSecrets来引用刚创建的myregistrykey