

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](#)。

```
1 $ docker login --username=fox666 registry.cn-hangzhou.aliyuncs.com
2 $ docker tag [ImageId] registry.cn-hangzhou.aliyuncs.com/fox666/tulingmall-product:[镜像版本号]
3 $ 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:

```
1 apiVersion: apps/v1
2 kind: Deployment
3 metadata:
4   name: tulingmall-product-deployment
5   labels:
6     app: tulingmall-product
7 spec:
8   replicas: 2
9   selector:
10    matchLabels:
11      app: tulingmall-product
12 template:
13   metadata:
14     labels:
15       app: tulingmall-product
16 spec:
17   hostNetwork: true # 主机模式, 初学者建议使用此模式
```

```

18     imagePullSecrets:
19     - name: myregistrykey #私有仓库的secret
20     containers:
21     - name: tulingmall-product
22       image: registry.cn-hangzhou.aliyuncs.com/fox666/tulingmall-product:0.0.5
23       imagePullPolicy: Always
24       ports:
25       - containerPort: 8866
26       env:
27       - name: TZ
28         value: Asia/Shanghai
29       - name: spring.cloud.nacos.config.server-addr
30         value: 192.168.65.174:8848
31       - name: LOG_FILE
32         value: /var/logs
33       volumeMounts:
34       - mountPath: /var/logs
35         name: log-volume
36     volumes:
37     - name: log-volume
38       hostPath:
39       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：

```

1 kubectl apply -f tulingmall-product-deployment.yaml
2
3 #查看部署pod的详细信息
4 kubectl describe pod xxxx

```

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

```
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: tulingmall-product-service
5  spec:
6    type: NodePort
7    selector:
8      app: tulingmall-product
9    ports:
10     - name: http
11       protocol: TCP
12       port: 8866
13       targetPort: 8866
```

执行如下命令创建商品服务的service:

```
1  kubectl apply -f tulingmall-product-service.yaml
2
3  #查看部署service的详细信息
4  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>

```
1  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

```
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: tulingmall-authcenter-deployment
5    labels:
6      app: tulingmall-authcenter
7  spec:
8    replicas: 1
9    selector:
10     matchLabels:
11       app: tulingmall-authcenter
12  template:
13    metadata:
14     labels:
15       app: tulingmall-authcenter
16    spec:
17     hostNetwork: true
18     imagePullSecrets:
19       - name: myregistrykey
20     containers:
21       - name: tulingmall-authcenter
22         image: registry.cn-hangzhou.aliyuncs.com/fox666/tulingmall-authcenter:0.0.5
23         imagePullPolicy: Always
24         ports:
25           - containerPort: 9999
26         env:
27           - name: TZ
28             value: Asia/Shanghai
29           - name: spring.cloud.nacos.config.server-addr
30             value: 192.168.65.174:8848
31           - name: LOG_FILE
32             value: /var/logs
33         volumeMounts:
34           - mountPath: /var/logs
35             name: log-volume
36     volumes:
37       - name: log-volume
38         hostPath:
```

```
39         path: /mydata/k8s-app/tulingmall-authcenter/logs
40         type: DirectoryOrCreate
41
42 ---
43 apiVersion: v1
44 kind: Service
45 metadata:
46   name: tulingmall-authcenter-service
47 spec:
48   type: NodePort
49   selector:
50     app: tulingmall-authcenter
51   ports:
52   - name: http
53     protocol: TCP
54     port: 9999
55     targetPort: 9999
56
```

tulingmall-gateway

```
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: tulingmall-gateway-deployment
5    labels:
6      app: tulingmall-gateway
7  spec:
8    replicas: 1
9    selector:
10      matchLabels:
11        app: tulingmall-gateway
12    template:
```

```
13     metadata:
14         labels:
15             app: tulingmall-gateway
16     spec:
17         hostNetwork: true
18         imagePullSecrets:
19             - name: myregistrykey
20         containers:
21             - name: tulingmall-gateway
22               image: registry.cn-hangzhou.aliyuncs.com/fox666/tulingmall-gateway:0.0.5
23               imagePullPolicy: Always
24               ports:
25                 - containerPort: 8888
26               env:
27                 - name: TZ
28                   value: Asia/Shanghai
29                 - name: spring.cloud.nacos.config.server-addr
30                   value: 192.168.65.174:8848
31                 - name: LOG_FILE
32                   value: /var/logs
33               volumeMounts:
34                 - mountPath: /var/logs
35                   name: log-volume
36         volumes:
37             - name: log-volume
38               hostPath:
39                 path: /mydata/k8s-app/tulingmall-gateway/logs
40                 type: DirectoryOrCreate
41
42 ---
43 apiVersion: v1
44 kind: Service
45 metadata:
46     name: tulingmall-gateway-service
47 spec:
48     type: NodePort
49     selector:
50         app: tulingmall-gateway
51     ports:
52         - name: http
```

```
53     protocol: TCP
54     port: 8888
55     targetPort: 8888
56
```

tulingmall-order-curr

```
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: tulingmall-order-curr-deployment
5    labels:
6      app: tulingmall-order-curr
7  spec:
8    replicas: 1
9    selector:
10     matchLabels:
11       app: tulingmall-order-curr
12  template:
13    metadata:
14     labels:
15       app: tulingmall-order-curr
16    spec:
17     hostNetwork: true
18     imagePullSecrets:
19     - name: myregistrykey
20     containers:
21     - name: tulingmall-order-curr
22       image: registry.cn-hangzhou.aliyuncs.com/fox666/tulingmall-order-curr:0.0.5
23       imagePullPolicy: Always
24       ports:
25       - containerPort: 8844
26       env:
27       - name: TZ
28         value: Asia/Shanghai
29       - name: spring.cloud.nacos.config.server-addr
```

```

30         value: 192.168.65.174:8848
31     - name: LOG_FILE
32       value: /var/logs
33     volumeMounts:
34     - mountPath: /var/logs
35       name: log-volume
36   volumes:
37   - name: log-volume
38     hostPath:
39       path: /mydata/k8s-app/tulingmall-order-curr/logs
40       type: DirectoryOrCreate
41
42   ---
43
44   apiVersion: v1
45   kind: Service
46   metadata:
47     name: tulingmall-order-curr-service
48   spec:
49     type: NodePort
50     selector:
51       app: tulingmall-order-curr
52     ports:
53     - name: http
54       protocol: TCP
55       port: 8844
56       targetPort: 8844

```

tulingmall-cart

```

1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: tulingmall-cart-deployment
5    labels:

```



```
6     app: tulingmall-cart
7 spec:
8     replicas: 1
9     selector:
10         matchLabels:
11             app: tulingmall-cart
12     template:
13         metadata:
14             labels:
15                 app: tulingmall-cart
16     spec:
17         hostNetwork: true
18         imagePullSecrets:
19             - name: myregistrykey
20         containers:
21             - name: tulingmall-cart
22               image: registry.cn-hangzhou.aliyuncs.com/fox666/tulingmall-cart:0.0.5
23               imagePullPolicy: Always
24               ports:
25                 - containerPort: 8855
26               env:
27                 - name: TZ
28                   value: Asia/Shanghai
29                 - name: spring.cloud.nacos.config.server-addr
30                   value: 192.168.65.174:8848
31                 - name: LOG_FILE
32                   value: /var/logs
33               volumeMounts:
34                 - mountPath: /var/logs
35                   name: log-volume
36         volumes:
37             - name: log-volume
38               hostPath:
39                 path: /mydata/k8s-app/tulingmall-cart/logs
40                 type: DirectoryOrCreate
41
42 ---
43 apiVersion: v1
44 kind: Service
45 metadata:
```

```
46   name: tulingmall-cart-service
47 spec:
48   type: NodePort
49   selector:
50     app: tulingmall-cart
51   ports:
52   - name: http
53     protocol: TCP
54     port: 8855
55     targetPort: 8855
56
```

tulingmall-unqid

```
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: tulingmall-unqid-deployment
5    labels:
6      app: tulingmall-unqid
7  spec:
8    replicas: 1
9    selector:
10     matchLabels:
11       app: tulingmall-unqid
12  template:
13    metadata:
14      labels:
15        app: tulingmall-unqid
16    spec:
17      hostNetwork: true
18      imagePullSecrets:
19      - name: myregistrykey
20      containers:
21      - name: tulingmall-unqid
22        image: registry.cn-hangzhou.aliyuncs.com/fox666/tulingmall-unqid:0.0.5
```

```
23     imagePullPolicy: Always
24     ports:
25       - containerPort: 8833
26     env:
27       - name: TZ
28         value: Asia/Shanghai
29       - name: spring.cloud.nacos.config.server-addr
30         value: 192.168.65.174:8848
31       - name: LOG_FILE
32         value: /var/logs
33     volumeMounts:
34       - mountPath: /var/logs
35         name: log-volume
36     volumes:
37       - name: log-volume
38         hostPath:
39           path: /mydata/k8s-app/tulingmall-unqid/logs
40           type: DirectoryOrCreate
41
42 ---
43 apiVersion: v1
44 kind: Service
45 metadata:
46   name: tulingmall-unqid-service
47 spec:
48   type: NodePort
49   selector:
50     app: tulingmall-unqid
51   ports:
52     - name: http
53       protocol: TCP
54       port: 8833
55       targetPort: 8833
56
```

tulingmall-member

```
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: tulingmall-member-deployment
5    labels:
6      app: tulingmall-member
7  spec:
8    replicas: 1
9    selector:
10     matchLabels:
11       app: tulingmall-member
12  template:
13    metadata:
14     labels:
15       app: tulingmall-member
16    spec:
17     hostNetwork: true
18     imagePullSecrets:
19       - name: myregistrykey
20     containers:
21       - name: tulingmall-member
22         image: registry.cn-hangzhou.aliyuncs.com/fox666/tulingmall-member:0.0.5
23         imagePullPolicy: Always
24         ports:
25           - containerPort: 8877
26         env:
27           - name: TZ
28             value: Asia/Shanghai
29           - name: spring.cloud.nacos.config.server-addr
30             value: 192.168.65.174:8848
31           - name: LOG_FILE
32             value: /var/logs
33         volumeMounts:
34           - mountPath: /var/logs
35             name: log-volume
36     volumes:
37       - name: log-volume
38         hostPath:
39           path: /mydata/k8s-app/tulingmall-member/logs
```

```
40         type: DirectoryOrCreate
41
42     ---
43     apiVersion: v1
44     kind: Service
45     metadata:
46       name: tulingmall-member-service
47     spec:
48       type: NodePort
49       selector:
50         app: tulingmall-member
51       ports:
52         - name: http
53           protocol: TCP
54           port: 8877
55           targetPort: 8877
56
```

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

```
1 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：

```
1 apiVersion: networking.k8s.io/v1
2 kind: Ingress
3 metadata:
4   name: tulingmall-gateway-ingress
5 spec:
```

```
6 rules:
7   - host: gateway.tuling.com
8     http:
9       paths:
10        - path: /
11          pathType: Prefix
12          backend:
13            service:
14              name: tulingmall-gateway-service
15              port:
16                number: 8888
17 ingressClassName: nginx
```

执行如下命令生效规则：

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

查看生效的ingress规则：

```
1 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)

```
1 192.168.65.137 gateway.tuling.com
```

配置完后直接在客户机用域名请求下网关：

GET

gateway.tuling.com/pms/productInfo/27

Send

ParamsAuthHeaders (7)BodyPre-req. TestsSettingsCookies

Query Params

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		

BodyCookiesHeaders (4)Test Results200 OK77 ms3.28 KBSave as Example

PrettyRawPreviewVisualizeJSON

```
1 {
2   "code": 200,
3   "message": "操作成功",
4   "data": {
5     "id": 27,
6     "brandId": 6,
7     "productCategoryId": 19,
8     "feightTemplateId": 0,
9     "productAttributeCategoryId": 3,
10    "name": "小米8 全面屏游戏智能手机 12GB+64GB 黑色 全网通4G 双卡双待",
11    "pic": "http://macro-oss.oss-cn-shenzhen.aliyuncs.com/mall/images/20180615/xiaomi.jpg",
12    "productSn": "7437788",
13    "deleteStatus": 0,
14    "publishStatus": 1
```

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

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

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

```
1 [ root@curl:/ ]$ nslookup kubernetes.default
2 Server:      10.96.0.10
3 Address 1:  10.96.0.10 kube-dns.kube-system.svc.cluster.local
4
5 Name:        kubernetes.default
```

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

```
7
```

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]

```
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: mysql-product
5  spec:
6    ports:
7      - name: mysql-product
8        port: 3306
9        protocol: TCP
10       targetPort: 3306
11       type: NodePort
12
13  ---
14  apiVersion: v1
15  kind: Endpoints
16  metadata:
17    name: mysql-product
18  subsets:
19    - addresses:
20      - ip: 192.168.65.71
21      ports:
22        - name: mysql-product
23          port: 3306
24          protocol: TCP
```

执行如下命令


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

测试：容器内部可以正常访问mysql-product.tulingmall

nacos

```
1 apiVersion: v1
2 kind: Service
3 metadata:
4   name: nacos
5 spec:
6   ports:
7     - port: 8848
8       name: nacos
9       targetPort: 8848
10    - port: 9848
11      name: client-rpc
12      targetPort: 9848
13    - port: 9849
14      name: raft-rpc
15      targetPort: 9849
16   type: NodePort
17
18 ---
19 apiVersion: v1
20 kind: Endpoints
21 metadata:
22   name: nacos
23 subsets:
24   - addresses:
25     - ip: 192.168.65.174
26     ports:
27     - port: 8848
28       name: nacos
```

```
29     - port: 9848
30     name: client-rpc
31     - port: 9849
32     name: raft-rpc
```

tulingmall-product

```
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: tulingmall-product-deployment
5    labels:
6      app: tulingmall-product
7  spec:
8    replicas: 2
9    selector:
10     matchLabels:
11       app: tulingmall-product
12  template:
13    metadata:
14     labels:
15       app: tulingmall-product
16    spec:
17     #hostNetwork: true
18     imagePullSecrets:
19     - name: myregistrykey
20     containers:
21     - name: tulingmall-product
22       image: registry.cn-hangzhou.aliyuncs.com/fox666/tulingmall-product:0.0.5
23       imagePullPolicy: Always
24       ports:
25       - containerPort: 8866
26       env:
27       - name: TZ
28         value: Asia/Shanghai
29       - name: spring.cloud.nacos.config.server-addr
30         value: nacos.default:8848
```

```
31         - name: LOG_FILE
32           value: /var/logs
33       volumeMounts:
34         - mountPath: /var/logs
35           name: log-volume
36       volumes:
37         - name: log-volume
38           hostPath:
39             path: /mydata/k8s-app/tulingmall-product/logs
40
41 ---
42
43 apiVersion: v1
44 kind: Service
45 metadata:
46   name: tulingmall-product-service
47 spec:
48   type: NodePort
49   selector:
50     app: tulingmall-product
51   ports:
52     - name: http
53       protocol: TCP
54       port: 8866
55       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：

```
1 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