
Kaikeba
开课吧

Kubernetes
服务发现及应用配置

@ 张晋涛

01 · K8S 中的 Service 和流量路由

02 · K8S 中的应用配置方式

03 · 总结

| 01 |

K8S 中的 Service 和流量路由

准备环境：创建两个 Deployment 资源

```
→  ~ kubectl create deploy redis-0 --image="ghcr.io/moelove/redis:alpine"
deployment.apps/redis-0 created
→  ~ kubectl create deploy redis-1 --image="ghcr.io/moelove/redis:alpine"
deployment.apps/redis-1 created
```

| 01-1 |

K8S 中 Pod 间通信

获取 Pod 的 IP

```
→  ~ kubectl get pod -l app=redis-0 --output jsonpath='{.items[0].status.podIP}'
10.244.1.9
→  ~ kubectl get pod -l app=redis-1 --output jsonpath='{.items[0].status.podIP}'
10.244.1.10
```

Pod 间通信

```
→  ~ kubectl exec deploy/redis-0 -- ping -c 1 `kubectl get pod -l app=redis-1 \
--output jsonpath='{.items[0].status.podIP}'`
PING 10.244.1.10 (10.244.1.10): 56 data bytes
64 bytes from 10.244.1.10: seq=0 ttl=63 time=0.087 ms

--- 10.244.1.10 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 0.087/0.087/0.087 ms
```

Pod 间通信

```
➜  ~ kubectl get pod -l app=redis-1 --output jsonpath='{.items[0].status.podIP}'
10.244.1.10
➜  ~ kubectl delete pods -l app=redis-1
pod "redis-1-5987df6b97-hpnt5" deleted
➜  ~ kubectl get pod -l app=redis-1 --output jsonpath='{.items[0].status.podIP}'
10.244.1.11
➜  ~ kubectl exec deploy/redis-0 -- ping -c 1 `kubectl get pod -l app=redis-1 --output
jsonpath='{.items[0].status.podIP}'`
PING 10.244.1.11 (10.244.1.11): 56 data bytes
64 bytes from 10.244.1.11: seq=0 ttl=63 time=0.072 ms

--- 10.244.1.11 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 0.072/0.072/0.072 ms
```

Pod 间通信

- 每个 Pod 都有自己的 IP
- Pod 间可直接通过 IP 通信
- Pod IP 是可变的
- Pod IP 一般情况下不能提前获取

```
➜  ~ kubectl expose deploy redis-1 --port=6379
service/redis-1 exposed
➜  ~ kubectl get svc -l app=redis-1
NAME      TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)    AGE
redis-1   ClusterIP   10.96.22.135   <none>        6379/TCP   10s
➜  ~ kubectl exec deploy/redis-0 -- ping -c 1 redis-1
PING redis-1 (10.96.22.135): 56 data bytes

--- redis-1 ping statistics ---
1 packets transmitted, 0 packets received, 100% packet loss
command terminated with exit code 1
```

K8S 中 Service 的类型

- ClusterIP
- NodePort
- LoadBalancer
- ExternalName

Service 的声明

- spec.ports: 端口定义
- spec.selector: 选择器
- spec.type: 类型

```yaml
apiVersion: v1
kind: Service
metadata:
  name: redis-1
spec:
  ports:
  - port: 6379
    protocol: TCP
    targetPort: 6379
  selector:
    app: redis-1
  type: ClusterIP
```

Pod 可通过 Service 通信

```
➜  ~ kubectl exec deploy/redis-0 -- sh -c 'nslookup redis-1 | grep "^[^*]"'
Server:         10.96.0.10
Address:        10.96.0.10:53
Name:   redis-1.default.svc.cluster.local
Address: 10.96.22.135
➜  ~ kubectl exec deploy/redis-0 -- sh -c 'redis-cli -h redis-1 -p 6379 ping'
PONG
```

| 01-2 |

外部流量与 Pod 间通信

外部流量访问 K8S 中的服务

- NodePort
- LoadBalancer

使用 NodePort 访问 Pod

```
➜  ~ kubectl expose deploy/redis-1 --port=6379 --name=redis-1-nodeport --type=NodePort
 service/redis-1-nodeport exposed
➜  ~ kubectl get svc -l app=redis-1
NAME               TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
redis-1            ClusterIP   10.96.22.135    <none>        6379/TCP         64m
redis-1-nodeport   NodePort    10.96.181.249   <none>        6379:32033/TCP   13m
➜  ~ kubectl get nodes  -o custom-columns=IP:status.addresses\[0\].address
IP
172.18.0.3
172.18.0.2
➜  ~ redis-cli -h 172.18.0.3 -p 32033 set name moelove
OK
➜  ~ redis-cli -h 172.18.0.2 -p 32033 get name
"moelove"
```

使用 LoadBalancer 访问 Pod

```
➜  ~ kubectl expose deploy/redis-1 --port=6379 --name=redis-1-lb --type=LoadBalancer
service/redis-1-lb exposed
➜  ~ kubectl get svc -l app=redis-1
NAME               TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
redis-1            ClusterIP      10.96.22.135    <none>        6379/TCP         72m
redis-1-lb         LoadBalancer   10.96.182.136   <pending>     6379:32369/TCP   6s
redis-1-nodeport   NodePort       10.96.181.249   <none>        6379:32033/TCP   20m
➜  ~ redis-cli -h 172.18.0.2 -p 32369 get name
"moelove"
```

| 01-3 |

Pod 与外部流量间通信

Pod 访问外部服务

- **ExternalName**

```
➜  ~ kubectl create service externalname httpbin --external-name httpbin.org
service/httpbin created
➜  ~ kubectl get svc -l app=httpbin
NAME      TYPE           CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
httpbin   ExternalName   <none>       httpbin.org   <none>    10s
➜  ~ kubectl exec deploy/redis-0 -- sh -c 'nslookup httpbin | grep "^[^*]"'
Server:         10.96.0.10
Address:        10.96.0.10:53
httpbin.default.svc.cluster.local       canonical name = httpbin.org
Name:   httpbin.org
Address: 34.199.75.4
Name:   httpbin.org
Address: 54.166.163.67
Name:   httpbin.org
Address: 34.231.30.52
Name:   httpbin.org
Address: 54.91.118.50
httpbin.default.svc.cluster.local       canonical name = httpbin.org
```

ExternalName 类型的 Service

- DNS CNAME
- 部分 HTTP 服务场景下受限 - 请求头限制
- Headless Service 是另一种可选项 - A

Headless Service

```
➜  ~ kubectl create service clusterip moelove-info --clusterip="None"
service/moelove-info created
➜  ~ kubectl apply -f moelove-ep.yaml
endpoints/moelove-info configured
➜  ~ kubectl exec deploy/redis-0 -- sh -c 'nslookup moelove-info | grep "^[^*]"'
Server:         10.96.0.10
Address:        10.96.0.10:53
Name:   moelove-info.default.svc.cluster.local
Address: 172.67.201.129
```

Service 的格式

- 例如： moelove-info.default.svc.cluster.local
- 格式： < 名称 >.< 命名空间 >.svc.cluster.<clusterDomain>

K8S 中 DNS 的说明

```
➜  ~ kubectl -n kube-system get svc
NAME       TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)                  AGE
kube-dns   ClusterIP   10.96.0.10   <none>        53/UDP,53/TCP,9153/TCP   8h
➜  ~ kubectl exec deploy/redis-0 -- cat /etc/resolv.conf
search default.svc.cluster.local svc.cluster.local cluster.local moelove.info
nameserver 10.96.0.10
options ndots:5
```

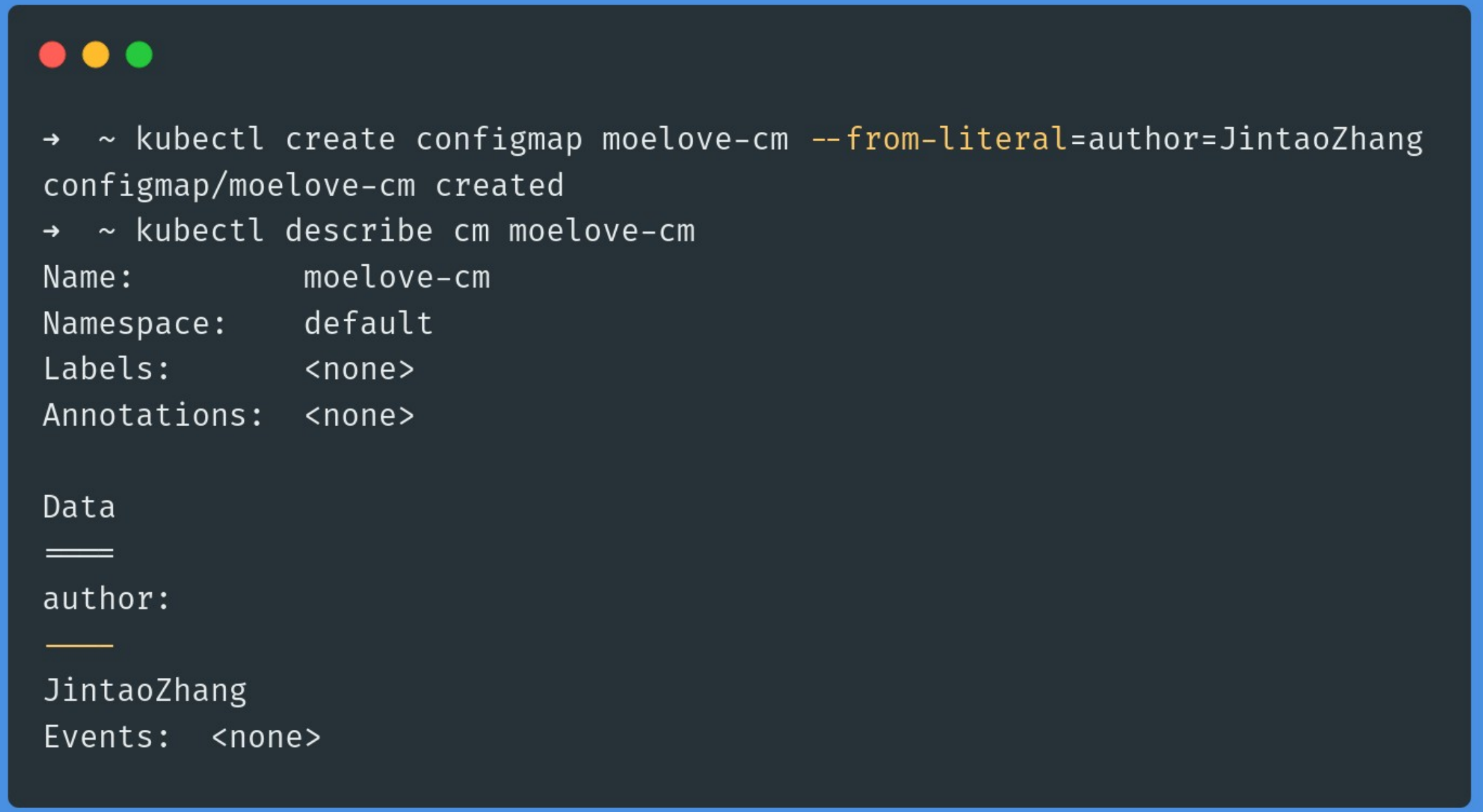
| 02 |

K8S 中的应用配置方式

K8S 中配置注入的主要方式

- ConfigMap
  - 保存一般配置，内容无加密
  - 可设置为环境变量
  - 可挂载为文件

- Secret
  - 通常保存“安全”要求高一些的配置，内容有基本编码
  - 可设置为环境变量
  - 可挂载为文件


→ ~ kubectl create configmap moelove-cm --from-literal=author=JintaoZhang
configmap/moelove-cm created
→ ~ kubectl describe cm moelove-cm
Name: moelove-cm
Namespace: default
Labels: <none>
Annotations: <none>
Data
====
author:

JintaoZhang
Events: <none>

```
→  ~ kubectl create secret generic moelove-secret --from-literal=author=JintaoZhang
secret/moelove-secret created
→  ~ kubectl describe secret moelove-secret
Name:         moelove-secret
Namespace:    default
Labels:       <none>
Annotations:  <none>

Type:  Opaque

Data
====
author:  11 bytes
```

```
➜  ~ kubectl get cm moelove-cm  -o yaml
apiVersion: v1
data:
  author: JintaoZhang
kind: ConfigMap
metadata:
  creationTimestamp: "2021-04-12T16:30:53Z"
  managedFields:
  - apiVersion: v1
    fieldsType: FieldsV1
    fieldsV1:
      f:data:
        .: {}
        f:author: {}
    manager: kubectl
    operation: Update
    time: "2021-04-12T16:30:53Z"
  name: moelove-cm
  namespace: default
  resourceVersion: "82948"
  uid: 5f7ab9b0-14cb-4e0e-8999-0f77626f0e1a
```

```
➜  ~ kubectl get secret moelove-secret -o yaml
apiVersion: v1
data:
  author: SmludGFvWmhhbmc=
kind: Secret
metadata:
  creationTimestamp: "2021-04-12T16:34:38Z"
  managedFields:
  - apiVersion: v1
    fieldsType: FieldsV1
    fieldsV1:
      f:data:
        .: {}
        f:author: {}
      f:type: {}
    manager: kubectl
    operation: Update
    time: "2021-04-12T16:34:38Z"
  name: moelove-secret
  namespace: default
  resourceVersion: "83349"
  uid: c724c583-c017-4037-9059-46d5a52d900c
type: Opaque
```

Secret 解码

```
➜  ~ kubectl get secret moelove-secret -o jsonpath="{.data.author}" |base64 -d
JintaoZhang
```

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: redis-3
  name: redis-3
spec:
  replicas: 1
  selector:
    matchLabels:
      app: redis-3
  template:
    metadata:
      labels:
        app: redis-3
    spec:
      containers:
      - image: ghcr.io/moelove/redis:alpine
        name: redis
        env:
        - name: AUTHOR
          valueFrom:
            configMapKeyRef:
              name: moelove-cm
              key: author
```

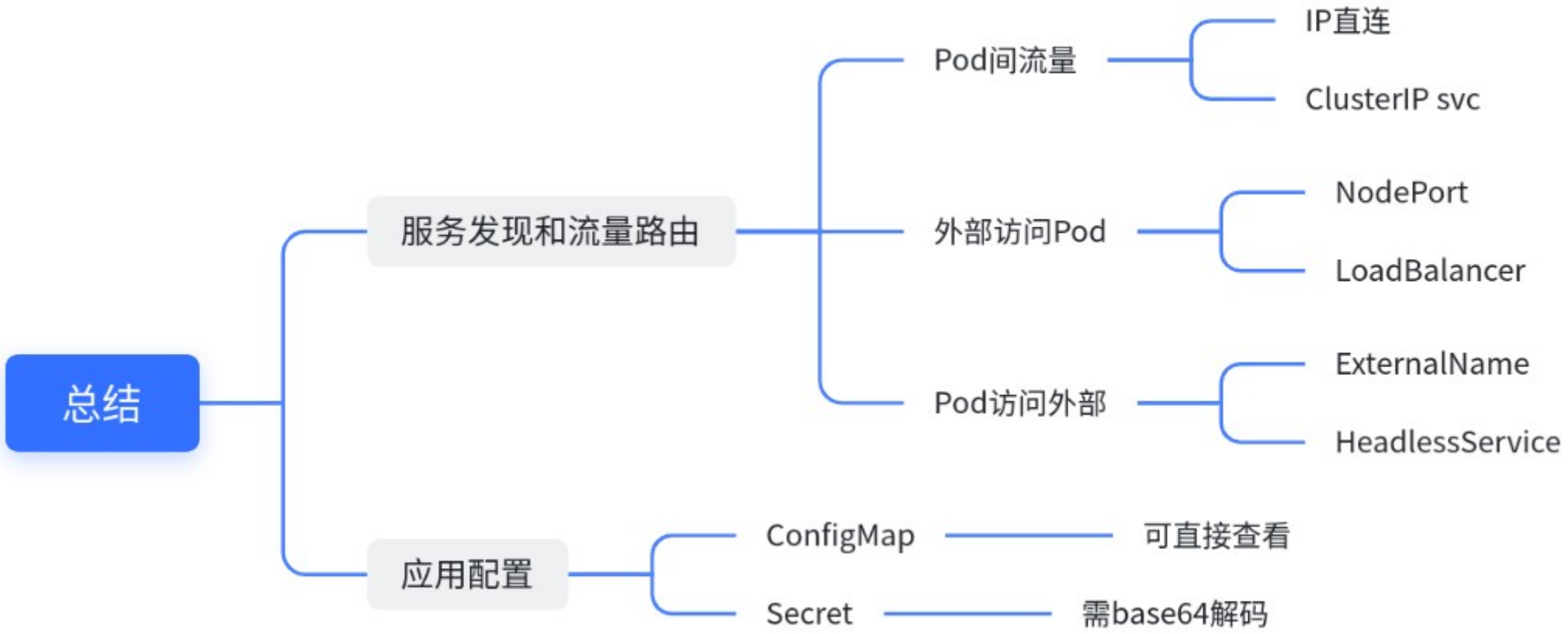
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: redis-4
  name: redis-4
spec:
  replicas: 1
  selector:
    matchLabels:
      app: redis-4
  template:
    metadata:
      labels:
        app: redis-4
    spec:
      containers:
      - image: ghcr.io/moelove/redis:alpine
        name: redis
        env:
        - name: AUTHOR
          valueFrom:
            secretKeyRef:
              name: moelove-secret
              key: author
```

验证

```
→  ~ kubectl apply -f redis-deploy.yaml
deployment.apps/redis-3 created
→  ~ kubectl apply -f redis-deploy-secret.yaml
deployment.apps/redis-4 created
→  ~ kubectl exec deploy/redis-3 -- printenv AUTHOR
JintaoZhang
→  ~ kubectl exec deploy/redis-4 -- printenv AUTHOR
JintaoZhang
```

03

总结


总结
服务发现和流量路由
Pod间流量
IP直连
ClusterIP svc
外部访问Pod
NodePort
LoadBalancer
Pod访问外部
ExternalName
HeadlessService
应用配置
ConfigMap
可直接查看
Secret
需base64解码

Reference

- Kubernetes 官网： https://kubernetes.io/
- KIND 官网： https://kind.sigs.k8s.io/
- 使用 KIND 搭建本地环境： https://zhuanlan.zhihu.com/p/105173589
- Kubernetes 上手实践： https://sourl.cn/UwcX5R

Kaikeba 开课吧

扫码关注开课吧

官方网站
www.kaikeba.com

联系电话
400-996-0826