TKE Ingress获取真实源IP Playbook指南(简化版)

2025-07-11 20:13

目录

背景

在TKE环境中，通过CLB七层负载均衡器获取真实源IP是常见需求。本Playbook详细指导如何**实现CLB非直连业务Pod**的方案，帮助您配置TKE Ingress以正确获取客户端真实源IP。

本指南为简化设计，跳过打相关docker镜像，使用我已推送到腾讯镜像仓库的的Flask镜像 `test-angel01.tencentcloudcr.com/kestrelli/kestrel-seven-real-ip:v1.0`，跳过镜像构建等步骤！

如有相关需求请访问：TKE Ingress获取真实源IP Playbook指南

前置条件

1. **腾讯云账号**：已开通容器服务(TKE)、云服务器(CVM)、容器镜像服务
2. **TKE集群**：版本≥1.14，已配置好kubectl访问凭证

快速开始

步骤1：创建Deployment

1.创建自定义命名空间（默认为default，自定义为kestrelli）

```
kubectl create ns kestrelli
```

2.创建 Deployment YAML 文件(workload.yaml)

```
apiVersion: apps/v1
kind: Deployment
metadata:
  # 修改工作负载名称，可换成自设
  name: kestrelli-real-ip
  # 增加命名空间
  namespace: kestrelli
spec:
  replicas: 2
  selector:
    matchLabels:
      app: kestrelli-real-ip          # 注意：selector 的标签也要同步修改，以匹配 template 中的标签
  template:
    metadata:
      labels:
        app: kestrelli-real-ip      # 修改 Pod 的标签，与 selector 保持一致
    spec:
      containers:
      - name: flask
        image: test-angel01.tencentcloudcr.com/kestrelli/kestrel-seven-real-ip:v1.0
        ports:
        - containerPort: 5000
```

📌 关键配置

- `metadata.labels` 需与后续 Service 选择器匹配
- `containerPort` 需与业务实际端口一致

3.部署工作负载

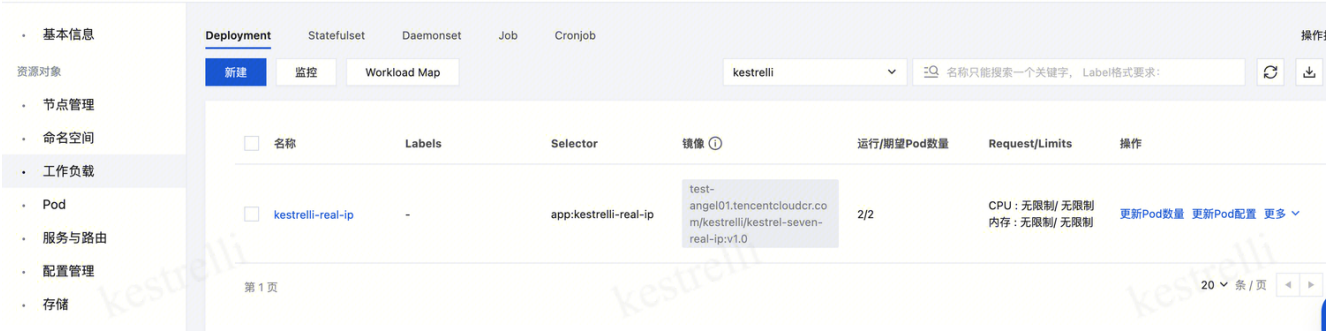
```
kubectl apply -f workload.yaml
```

4.验证 Pod 状态

```
#命名空间换成自己的
kubectl get pods -l app=kestrelli-real-ip -n kestelli
```

预期输出：✅ 看到2个`Running`状态的Pod

```
[root@VM-35-89-tlinux ~]# kubectl get pods -l app=kestrelli-real-ip -n kestrelli
NAME                                READY   STATUS    RESTARTS   AGE
kestrelli-real-ip-74c75b8cf4-7wn6f   1/1     Running   0          3m29s
kestrelli-real-ip-74c75b8cf4-xnvp8   1/1     Running   0          3m29s
```



步骤2：创建Service（NodePort类型）

1.创建 Service YAML 文件(svc.yaml)

```
apiVersion: v1
kind: Service
metadata:
  name: real-ip-svc
spec:
  selector:
    app: kestrelli-real-ip
  ports:
    - protocol: TCP
      port: 80         # 外部访问端口
      targetPort: 5000  # 映射到Flask的5000端口
  type: NodePort       # 非直连模式必需
```

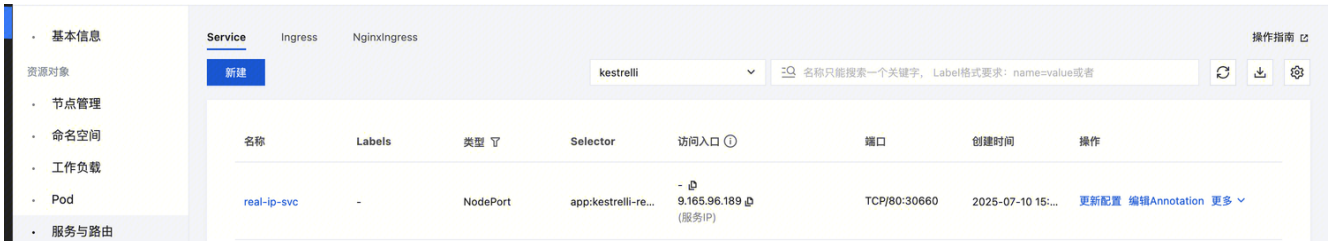
2.部署 Service

```
#指定命名空间（不指定为default）
kubectl apply -f svc.yaml -n kestrelli
```

3.验证 Service 配置

```
#工作负载指定的命名空间（这里为kestrelli）
kubectl describe svc real-ip-svc -n kestrelli
```

```
[root@VM-35-89-tlinux ~]# kubectl describe svc real-ip-svc -n kestrelli
Name:                     real-ip-svc
Namespace:                kestrelli
Labels:                   <none>
Annotations:              service.cloud.tencent.com/sync-begin-time: 2025-07-10T15:04:48+08:00
                          service.cloud.tencent.com/sync-end-time: 2025-07-10T15:04:48+08:00
Selector:                 app=kestrelli-real-ip
Type:                     NodePort
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       9.165.96.189
IPs:                      9.165.96.189
Port:                     <unset>  80/TCP
TargetPort:               5000/TCP
NodePort:                 <unset>  30660/TCP
Endpoints:                10.0.35.58:5000,10.0.35.46:5000
Session Affinity:         None
External Traffic Policy:  Cluster
Internal Traffic Policy:  Cluster
Events:
  Type    Reason                Age   From                Message
  ----    ------                ----  ----                -------
  Normal  EnsureServiceSuccess  3m32s  service-controller  Sync Success. ReturnCode: S2000
```



验证：

```
kubectl get svc real-ip-svc -n kestrelli
```

✅ 查看`PORT(S)`列显示 `80:3xxxx/TCP`（3xxxx为自动分配的节点端口）

```
[root@VM-35-89-tlinux ~]# kubectl get svc real-ip-svc -n kestrelli
NAME          TYPE       CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE
real-ip-svc   NodePort   9.165.96.189   <none>        80:30660/TCP   7m49s
```

步骤3：创建Ingress（核心配置）

1.创建 Ingrss YAML 文件（ingress.yaml）

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: real-ip-ingress
spec:
  ingressClassName: qcloud
  rules:
  - http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: real-ip-svc
  # 关联上一步的Service
            port:
              number: 80
```

2.部署 Ingress

```bash
#指定命名空间
kubectl apply -f ingress.yaml  -n kestrelli
```

3.获取访问地址：

```bash
#指定命名空间
kubectl get ingress real-ip-ingress -n kestrelli -o
jsonpath='{.status.loadBalancer.ingress[0].ip}'
```

```
[root@VM-35-89-tlinux ~]# kubectl get ingress real-ip-ingress -n kestrelli -o jsonpath='{.status.loadBalancer.ingress[0].ip}'
119.29.51.228[root@VM-35-89-tlinux ~]#
```

步骤4：验证真实源IP

执行命令：

```bash
curl http://<上一步获取的IP>
```

预期成功输出：

```
{
  "headers": {
    "X-Forwarded-For": "您的公网IP",
    "X-Real-Ip": "您的公网IP"
  }
}
```

```
119.29.51.228[root@VM-35-89-tlinux ~]# curl 119.29.51.228
{"headers":{"Accept":"*/*","Connection":"keep-alive","Host":"119.29.51.228","User-Agent":"curl/7.61.1","X-Client-Proto":"http","X-Client-Proto-Ver":"HTTP/1.1","X-Forwarded-For":"106.53.86.214","X-Forwarded-Proto":"http","X-Real-Ip":"106.53.86.214","X-Stgw-Time":"1752133041.129"},"message":"Here are your request headers","method":"GET"}
```

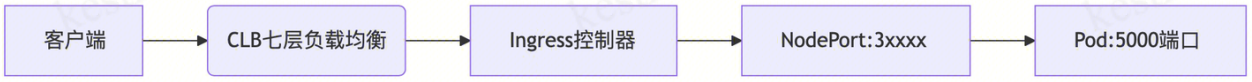
故障排查表

问题现象	解决方案
`curl`无响应	1. 检查Ingress IP是否正确 2. 执行 `kubectl describe ingress real-ip-ingress -n kestrelli（指定的命名空间）` 查看events
返回404错误	检查Service名称是否拼写正确（`real-ip-svc`）
看到Node IP而非公网IP	确认Ingress注解 `ingressClassName: qcloud` 已配置
镜像拉取失败	在集群所在VPC执行： `docker pull test-angel01.tencentcloudcr.com/kestrelli/kestrel-seven-real-ip` 测试网络连通性

> 💡 **锦囊**：所有YAML已通过测试，直接复制粘贴即可运行

原理解析

流量路径：



关键设计：

1. 镜像直接处理请求，返回`X-Forwarded-For`和`X-Real-IP`头,获取客户端真实源IP
2. Service的`NodePort`模式自动透传源IP
3. Ingress注解`qcloud`启用腾讯云CLB七层转发