TKE Ingress获取真实源IP Playbook指南

2025-07-11 19:52

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。

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
3. **开发环境**：Python环境(用于Flask应用)

快速开始

1. 创建工作负载(Deployment)

目标：创建初始Nginx工作负载作为基础

```
# 部署Nginx示例
kubectl create deployment nginx-demo --image=nginx:latest -n default
```

验证：

```
kubectl get pods -n default -w
```

```
[root@VM-17-154-tlinux ~]# kubectl get pods -n default -w
NAME                          READY   STATUS    RESTARTS   AGE
nginx-demo-5db5fb4b9b-fjjw5   1/1     Running   0          7m14s
```

2. 配置Service与Ingress

目标：创建Service暴露服务，并通过Ingress配置路由规则
1.创建 Service YAML 文件（svc.yaml）

```yaml
apiVersion: v1
kind: Service
metadata:
  name: nginx-svc
spec:
  ports:
  - port: 80
    targetPort: 80
  selector:
    app: nginx-demo
  type: NodePort
```

2.部署 Service

```
kubectl apply -f svc.yaml
```

3.验证 Service 配置

```
kubectl describe svc nginx-svc
```

```
[root@VM-17-154-tlinux ~]# kubectl describe svc nginx-svc
Name:                     nginx-svc
Namespace:                default
Labels:                   <none>
Annotations:              service.cloud.tencent.com/sync-begin-time: 2025-07-09T18:16:51+08:00
                          service.cloud.tencent.com/sync-end-time: 2025-07-09T18:16:51+08:00
Selector:                 app=nginx-demo
Type:                     NodePort
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       172.18.85.174
IPs:                      172.18.85.174
Port:                     <unset>  80/TCP
TargetPort:               80/TCP
NodePort:                 <unset>  31493/TCP
Endpoints:                10.15.17.147:80
Session Affinity:         None
External Traffic Policy:  Cluster
Internal Traffic Policy:  Cluster
Events:
  Type    Reason                Age   From                Message
  ----    ------                ----  ----                -------
  Normal  EnsureServiceSuccess  7m7s  service-controller  Sync Success. ReturnCode: S2000
```

4.创建 Ingrss YAML 文件（ingress.yaml）

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: demo-ingress
spec:
  ingressClassName: nginx
  rules:
  - host: example.com
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: nginx-svc
            port:
              number: 80
```

5.部署 Ingress

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

3. 准备构建环境(OrcaTerm)

目标：登录云服务器准备Docker构建环境

1. 进入云服务器控制台，选择CentOS 7.x实例
2. 点击"登录" → 选择"免密连接(TAT)"
3. 安装Docker环境：

```
sudo yum install -y yum-utils device-mapper-persistent-data lvm2
sudo yum-config-manager --add-repo
https://download.docker.com/linux/centos/docker-ce.repo
sudo yum install -y docker-ce docker-ce-cli containerd.io
sudo systemctl start docker && sudo systemctl enable docker
```

4. 创建Flask应用

目标：构建能显示请求头的Flask应用
新建一个文件夹（如kestrelli）
cd kestelli（你创建的文件夹）
创建一个名为`app.py`的文件，包含以下内容：

```python
from flask import Flask, request, jsonify
import logging

# 配置日志记录
logging.basicConfig(
    level=logging.INFO,
    format='%(asctime)s - %(levelname)s - %(message)s',
    handlers=[
        logging.StreamHandler(),
        logging.FileHandler('app.log')
    ]
)

app = Flask(__name__)

@app.route('/', methods=['GET', 'POST', 'PUT', 'DELETE', 'PATCH'])
def handle_request():
    """处理所有HTTP方法请求，打印并返回请求头"""
    # 获取所有请求头
    headers = dict(request.headers)

    # 打印请求头到控制台和日志文件
    logging.info("Received request with headers:")
    for header, value in headers.items():
        logging.info(f"{header}: {value}")

    # 返回JSON格式的请求头
    return jsonify({
        "message": "Here are your request headers",
        "headers": headers,
        "method": request.method
    })

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000, debug=True)
```

创建一个`requirements.txt`文件，列出所有需要的Python包：

```
Flask==2.3.2
gunicorn==20.1.0
```

创建一个`Dockerfile`用于容器化部署：

```dockerfile
# 使用官方Python运行时作为基础镜像
FROM python:3.9-slim

# 设置工作目录
WORKDIR /app

# 复制依赖文件并安装
COPY requirements.txt .
RUN pip install --no-cache-dir -r requirements.txt

# 复制应用代码
COPY app.py .

# 暴露端口
EXPOSE 5000

# 运行应用
CMD ["gunicorn", "--bind", "0.0.0.0:5000", "app:app"]
```

5. 构建推送镜像

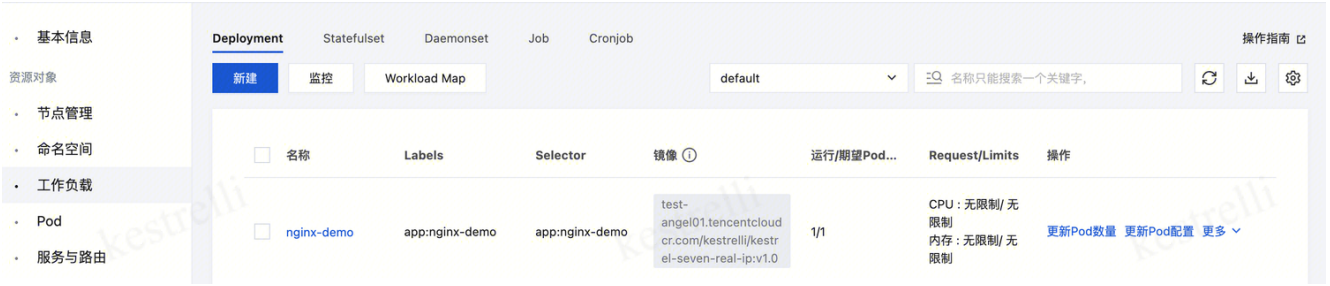
目标：将Flask应用容器化并推送到腾讯云镜像仓库

```bash
# 构建镜像
docker build -t real-ip-app .

# 登录镜像仓库（根据自己构建的镜像仓库指令操作，如我的所示）
docker login test-angel01.tencentcloudcr.com --username=10000xxxxxx --password=xxxxxx

# 标记
docker tag 00bc6bf8b412 test-angel01.tencentcloudcr.com/kestrelli/kestrel-seven-real-ip:v1.0

# 推送
docker push test-angel01.tencentcloudcr.com/kestrelli/kestrel-seven-real-ip:v1.0
```

6. 更新工作负载配置

目标：使用新镜像替换Nginx，并修改端口配置
1.修改Deployment镜像：

```
kubectl set image deployment/nginx-demo nginx=ccr.ccs.tencentyun.com/your-namespace/real-ip-app:v1 -n default
#或者直接使用我推送到镜像仓库的镜像
test-angel01.tencentcloudcr.com/kestrelli/kestrel-seven-real-ip
#使用指令为
kubectl set image deployment/nginx-demo nginx=test-angel01.tencentcloudcr.com/kestrelli/kestrel-seven-real-ip:v1.0 -n default
```



2.更新Service端口映射：

```
# 修改svc.yaml
ports:
- port: 80
  targetPort: 5000 # 修改为Flask监听端口
```

7. 验证真实源IP

目标：测试获取真实客户端IP

```
# 获取Ingress访问地址
kubectl get ingress real-ip-ingress -n default
# 测试访问
curl ADDRESS
```

预期输出：

```
{
  "x-forwarded-for": "客户端真实IP",
  "x-real-ip": "客户端真实IP"
}
```

```
[root@VM-17-154-tlinux ~]# kubectl get ingress real-ip-ingress -n default
NAME              CLASS    HOSTS   ADDRESS           PORTS   AGE
real-ip-ingress   <none>   *       193.112.231.145   80      12m
```

```
[root@VM-17-154-tlinux ~]# curl 193.112.231.145
{"headers":{"Accept":"*/*","Connection":"keep-alive","Host":"193.112.231.145","User-Agent":"curl/7.61.1","X-Client-Proto":"http",
"X-Client-Proto-Ver":"HTTP/1.1","X-Forwarded-For":"106.55.163.108","X-Forwarded-Proto":"http","X-Real-Ip":"106.55.163.108","X-Stg
w-Time":"1752067622.583"},"message":"Here are your request headers","method":"GET"}
```

常见问题

Q1: 为什么看不到真实IP？

可能原因：

1. Service未正确配置targetPort
2. Ingress注解未启用源IP保留
3. 防火墙/安全组拦截

解决方案：

```
kubectl annotate ingress <INGRESS_NAME> \
  kubernetes.io/ingress.existLbId=<CLB_ID> \
  kubernetes.io/ingress.subnetId=<SUBNET_ID> \
  --overwrite
```

Q2: 镜像推送失败如何处理？

1. 检查镜像仓库权限
2. 确认命名空间存在
3. 使用--debug参数查看详细错误：

```
docker push --debug ccr.ccs.tencentyun.com/your-namespace/real-ip-app:v1
```

Q3: 非直连模式与直连模式区别？

特性	GR模式	直连模式
网络路径	CLB→NodePort→Pod	CLB→Pod
源IP保留	需特殊配置	自动保留
适用场景	非生产环境	生产环境