TKE Ingress 获取真实源 IP（eni模式实现clb非直连业务pod）

2025-07-03 15:55

使用 TKE Ingress 获取真实源 IP

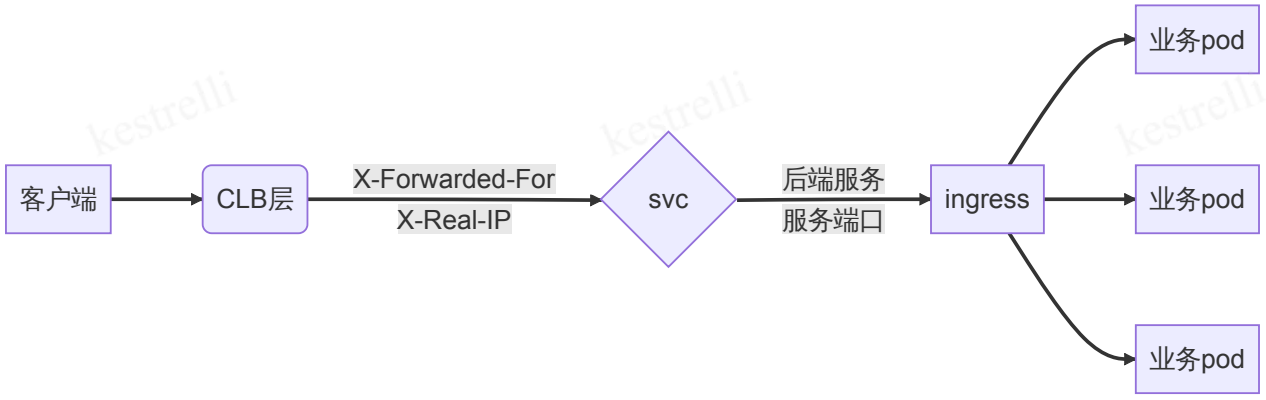
网络模式为VPC-CNI

如图所示

业务访问链路：

客户端-> LB类型svc->LB类型ingress -> nodport -> 业务pod

转发流程如图所示

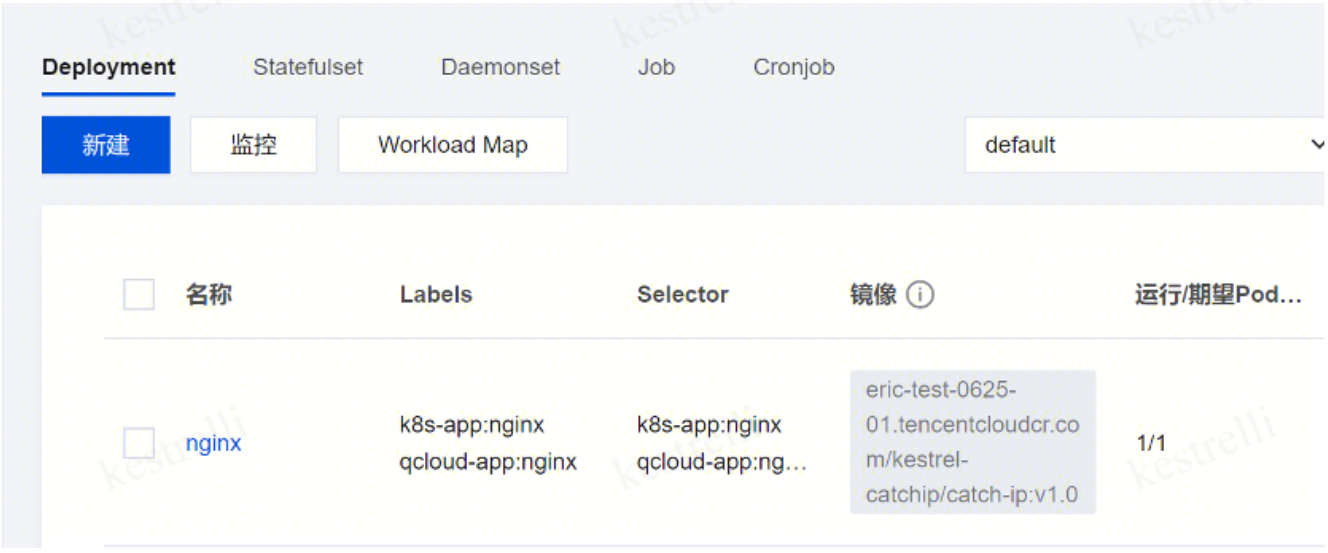


腾讯云负载均衡器（CLB 七层） 默认会将客户端真实源 IP 放至 HTTP Header 的 X-Forwarded-For 和 X-Real-IP 字段。当服务流量在经过 Service 四层转发后会保留上述字段，后端通过 Web 服务器代理配置或应用代码方式获取到客户端真实源 IP

通过容器服务控制台 获取源 IP 步骤如下：

1.创建一个工作负载（Deployment）

镜像使用nginx官方镜像，后期可以直接从镜像仓库中推送的镜像及对应版本进行替换



此时保证nginx的运行和期望pod的数量满足预期或在pod管理中保证状态为运行（running）

2．在服务与路由中调整service配置和创建Ingress

后期可根据Dockerfile文件中的暴露端口号进行容器端口替换

转发配置中，后端服务选择service的服务名和对应服务端口

3.登录腾讯云OrcaTerm

选择合适的云服务器进行登录

检查腾讯云OrcaTerm中是否存在docker，如果存在，需卸载旧版本docker

```
sudo yum remove docker \
    docker-client \
    docker-client-latest \
    docker-common \
    docker-latest \
    docker-latest-logrotate \
    docker-logrotate \
    docker-engine
```

如果不存在docker，需要安装docker

安装Docker的步骤如下：

```
安装必要的依赖包：
sudo yum install -y yum-utils device-mapper-persistent-data lvm2
添加Docker官方仓库：
sudo yum-config-manager --add-repo
https://download.docker.com/linux/centos/docker-ce.repo
安装Docker引擎：
sudo yum install -y docker-ce docker-ce-cli containerd.io
启动Docker服务并设置开机自启：
sudo systemctl start docker
sudo systemctl enable docker
```

4.创建一个完整的Web服务

创建一个app.py的文件，包含应用代码：

```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
```

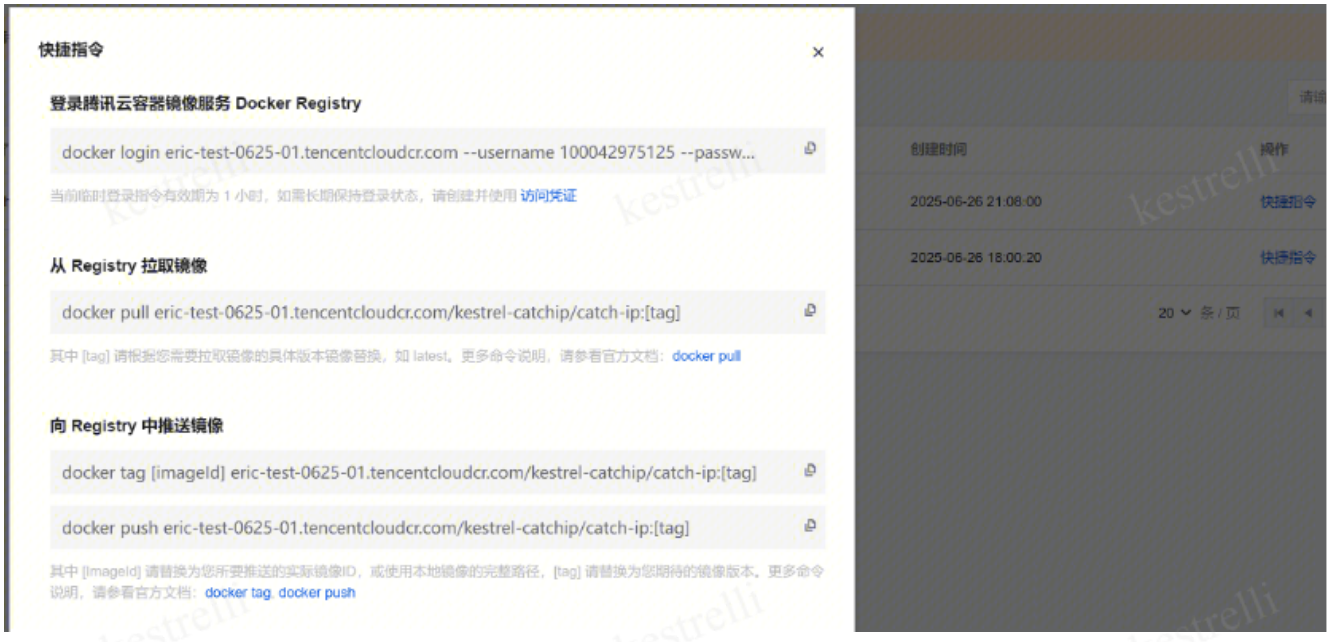
创建一个Dockerfile用于容器化部署：

```
# 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.Docker部署及镜像推送

构建Docker镜像：docker build -t flask-headers .

此时docker镜像已经完成构建，按照快捷指令推送镜像到腾讯云仓库



按指令操作完成镜像推送

```
[root@VM-16-20-tencentos ~]# docker tag  c37a5a2efd24  eric-test-0625-01.tencentcloudcr.com/kestrel-catchip/catch-ip:v1.0
[root@VM-16-20-tencentos ~]# docker push eric-test-0625-01.tencentcloudcr.com/kestrel-catchip/catch-ip:v1.0
The push refers to repository [eric-test-0625-01.tencentcloudcr.com/kestrel-catchip/catch-ip]
767616119c19: Pushed
275caaa2a12a: Pushed
ee1c98146918: Pushed
f2125cd07823: Pushed
f0d3b09df3b4: Pushed
b9a50396677d: Pushed
be76169c9d2c: Pushed
7fb72a7d1a8e: Pushed
v1.0: digest: sha256:ac4dde76727c9e04987bfd03225e34bbdb2f8e946e317ee8284a3abcd37e36b4 size: 1990
```

版本号如图

6.修改service及工作负载配置

根据推送到镜像仓库的镜像和对应的版本进行工作负载中更新Deployment的pod配置

7. 通过TKE Ingress 获取真实源 IP

在服务与路由中根据Ingress中的后端服务对应超链接点击访问，通过X-Forwarded-For或X-Real-Ip获取客户端真实IP