## 部署微服务系统

准备基础环境

mkdir -p /etc/yum.repos.d/backup

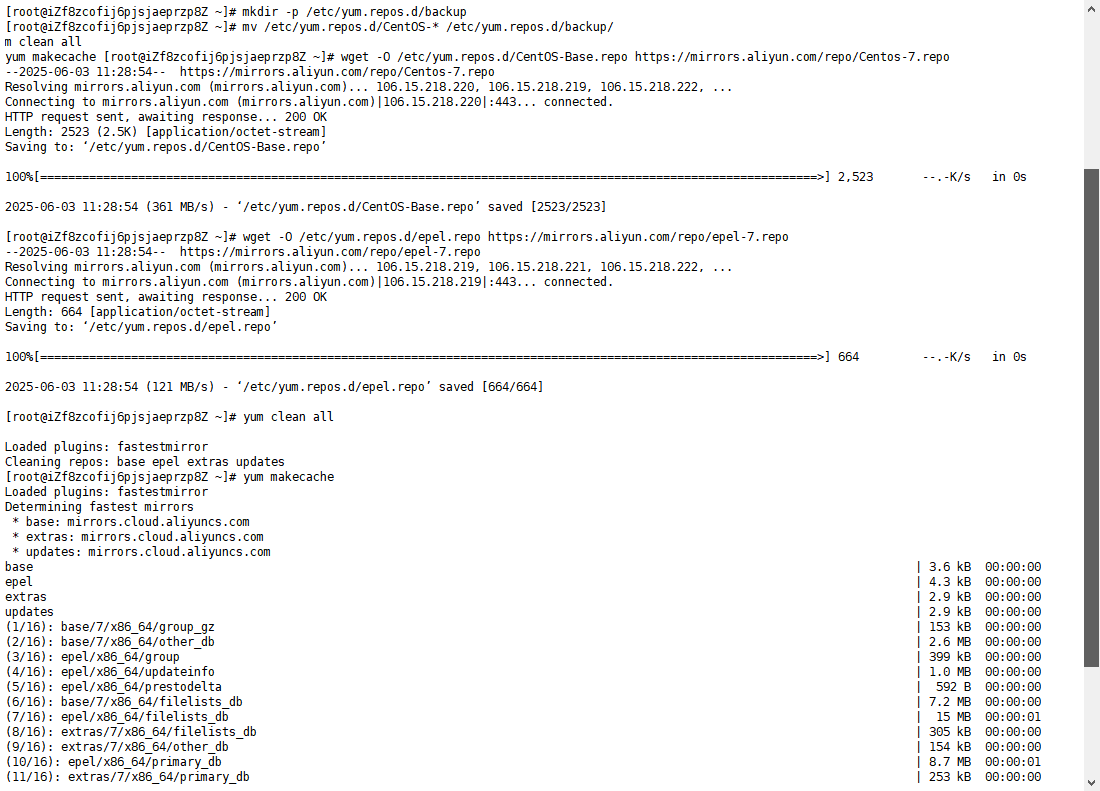
mv /etc/yum.repos.d/CentOS-\* /etc/yum.repos.d/backup/

wget -O /etc/yum.repos.d/CentOS-Base.repo https://mirrors.aliyun.com/repo/Centos-7.repo

wget -O /etc/yum.repos.d/epel.repo https://mirrors.aliyun.com/repo/epel-7.repo

yum clean all

yum makecache



yum install -y yum-utils device-mapper-persistent-data lvm2 wget vim lrzsz telnet

hostnamectl set-hostname k8s-master

systemctl stop firewalld

systemctl disable firewalld

setenforce 0

sed -i 's/=enforcing/=disabled/g' /etc/selinux/config

swapoff -a && sed -i '/swap/s/^/#/' /etc/fstab

cat > /etc/sysctl.d/k8s.conf << EOF

net.bridge.bridge-nf-call-ip6tables = 1

net.bridge.bridge-nf-call-iptables = 1

net.ipv4.ip\_forward = 1

vm.swappiness = 0

EOF

modprobe br\_netfilter && sysctl -p /etc/sysctl.d/k8s.conf

cat > /etc/sysconfig/modules/ipvs.modules <<EOF

#!/bin/bash

ipvs\_mods\_dir="/usr/lib/modules/\$(uname -r)/kernel/net/netfilter/ipvs"

for i in \$(ls \$ipvs\_mods\_dir | grep -o "^[^.]\*"); do

/sbin/modinfo -F filename \$i &>/dev/null

if [ \$? -eq 0 ]; then

/sbin/modprobe \$i

fi

done

EOF

chmod +x /etc/sysconfig/modules/ipvs.modules

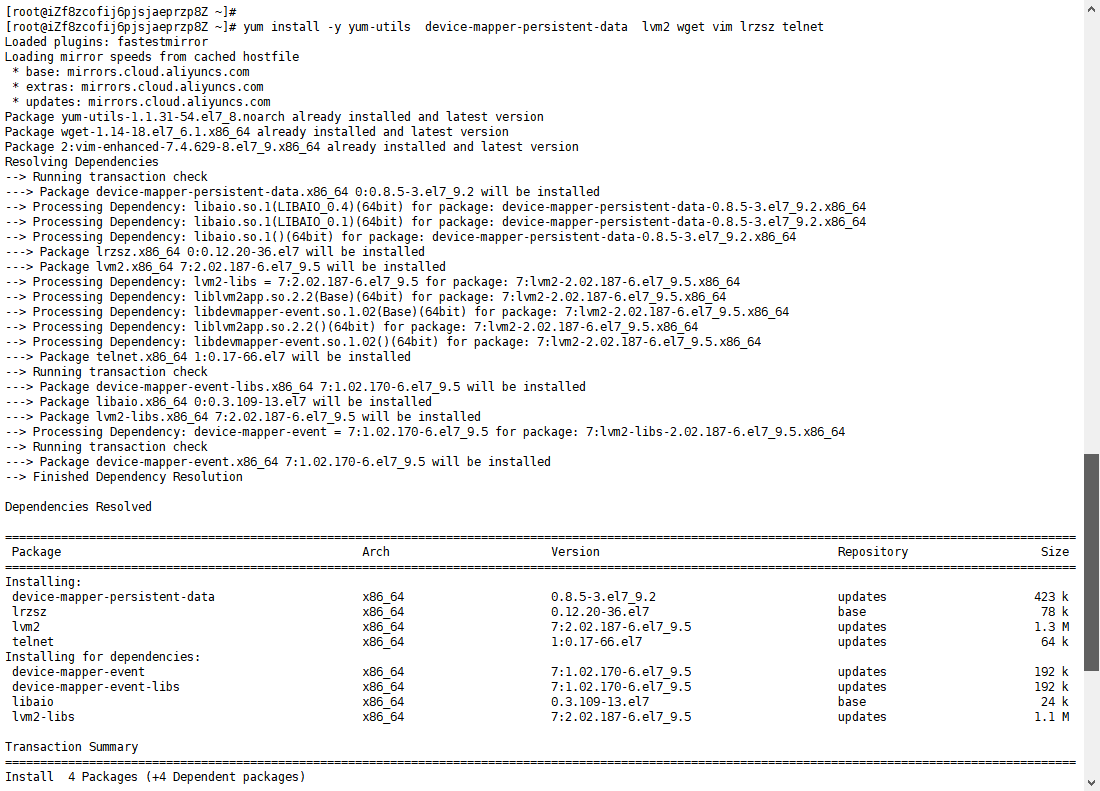
bash /etc/sysconfig/modules/ipvs.modules

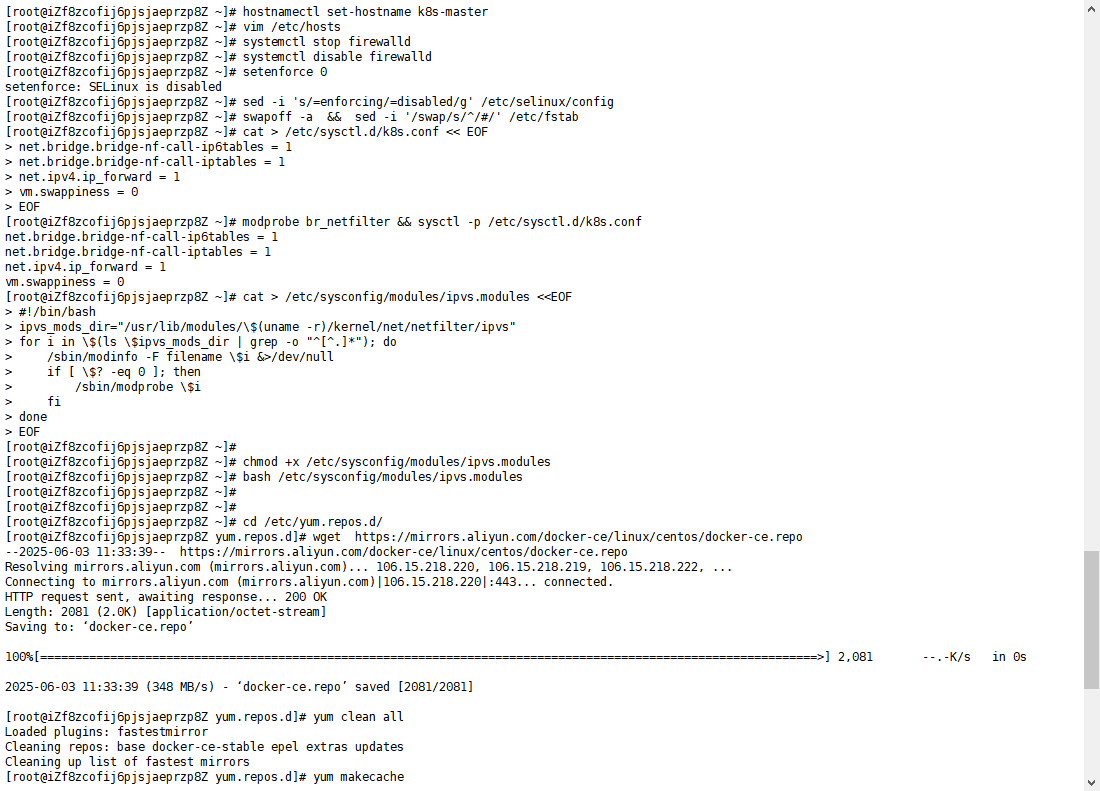
cd /etc/yum.repos.d/

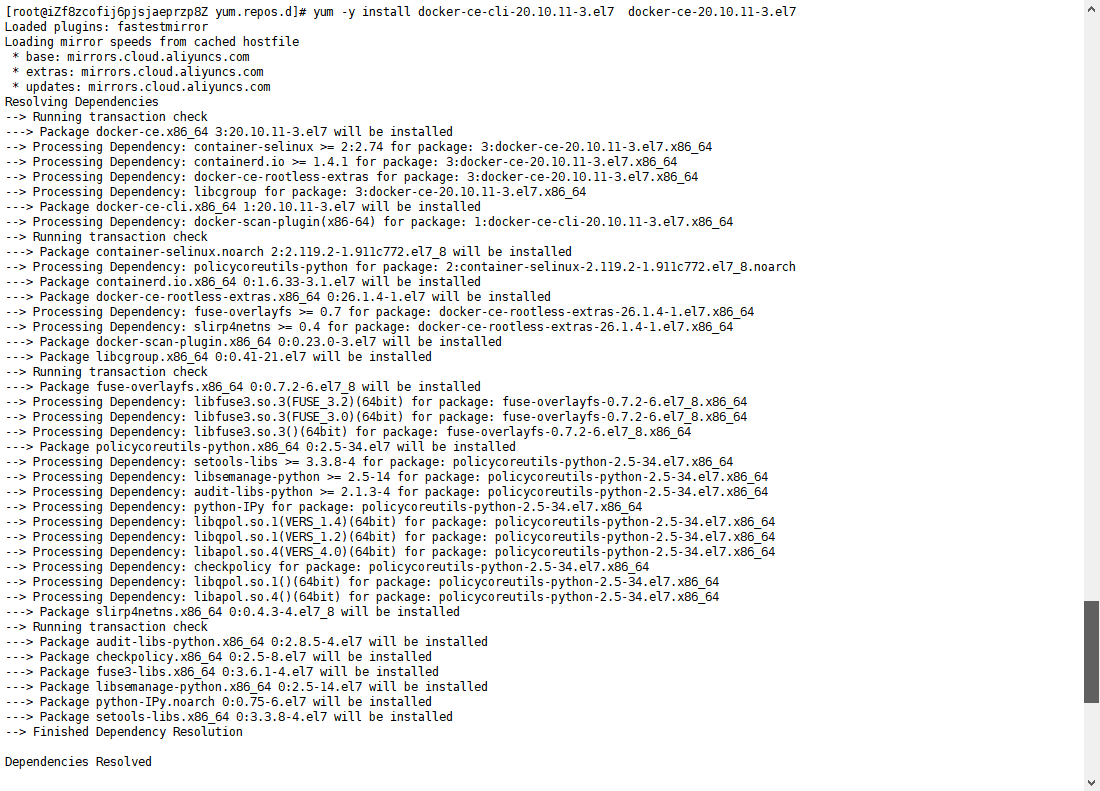
wget https://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo

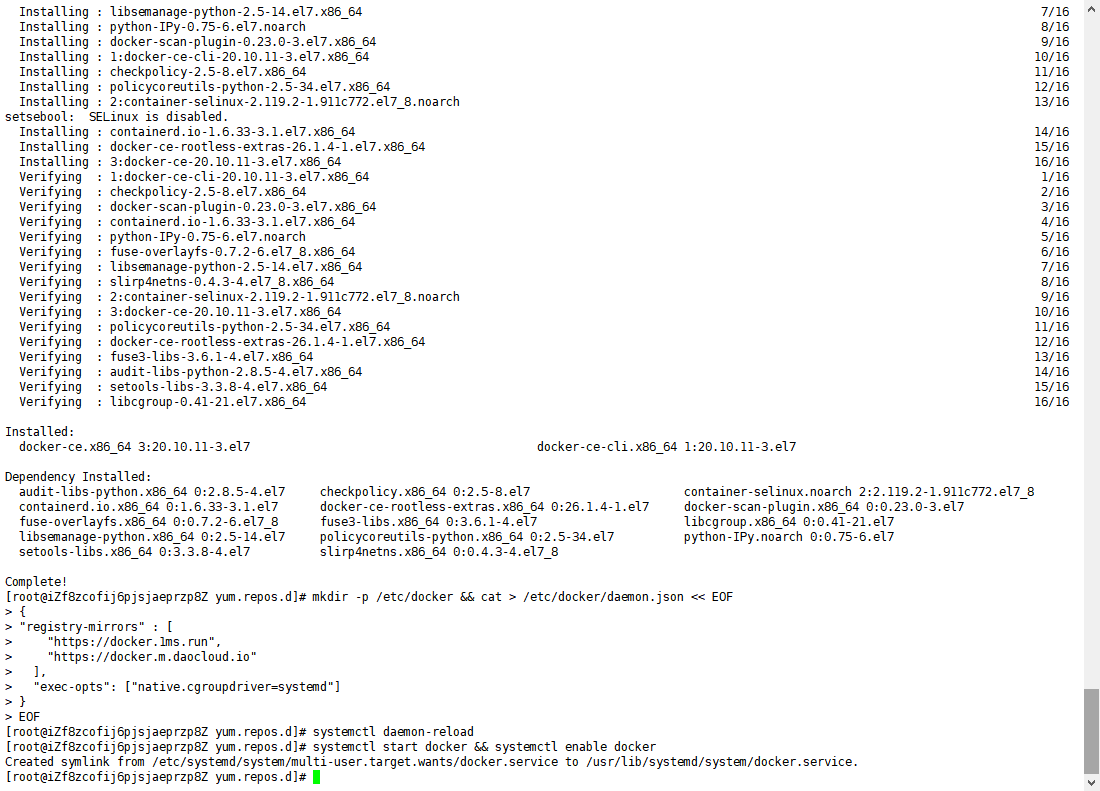
yum clean all

yum makecache





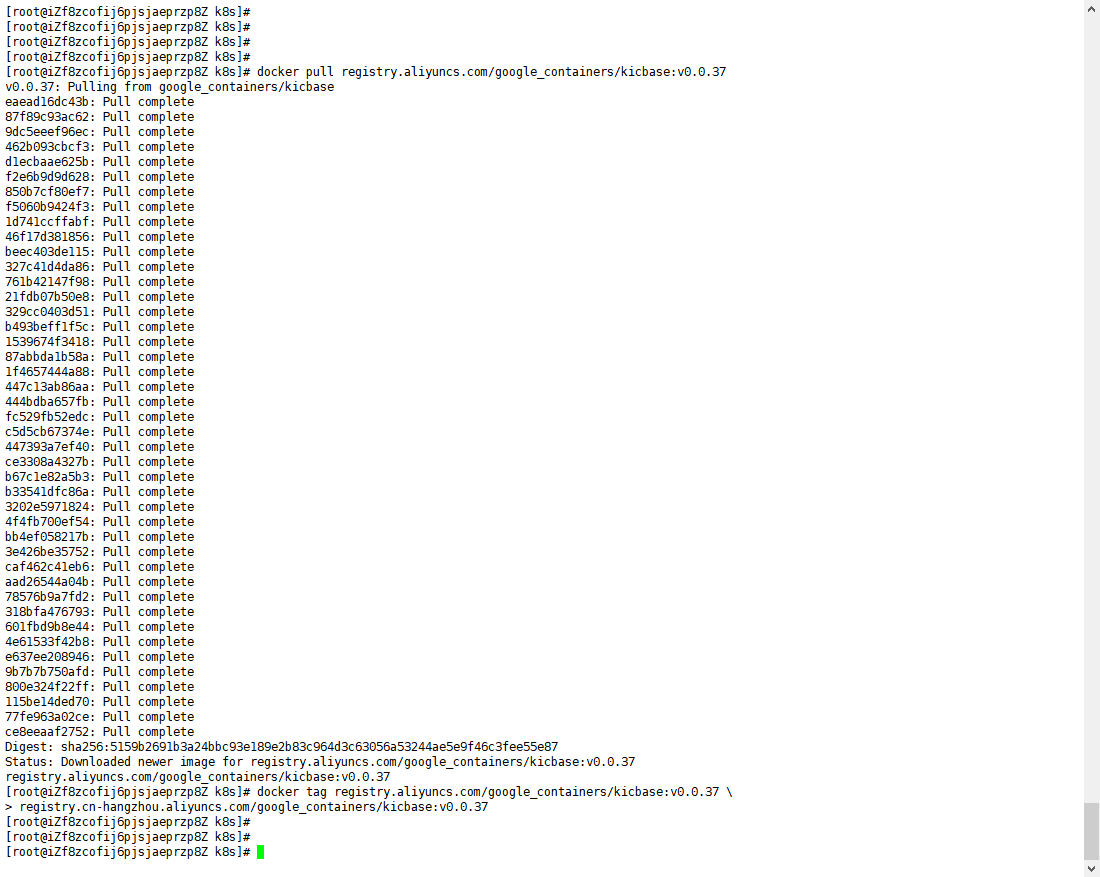




docker pull registry.aliyuncs.com/google\_containers/kicbase:v0.0.37

docker tag registry.aliyuncs.com/google\_containers/kicbase:v0.0.37 \

registry.cn-hangzhou.aliyuncs.com/google\_containers/kicbase:v0.0.37



mkdir /data/k8s && cd /data/k8s

curl -LO https://storage.googleapis.com/minikube/releases/v1.29.0/minikube-linux-amd64 && sudo install minikube-linux-amd64 /usr/local/bin/minikube

minikube start --memory=5000mb --force \

--driver=docker \

--image-mirror-country='cn' \

--kubernetes-version=v1.23.8 \

--base-image=registry.aliyuncs.com/google\_containers/kicbase:v0.0.37 \

--image-repository=registry.aliyuncs.com/google\_containers \

--registry-mirror="https://gcr.m.daocloud.io" \

--registry-mirror="https://registry.docker-cn.com" \

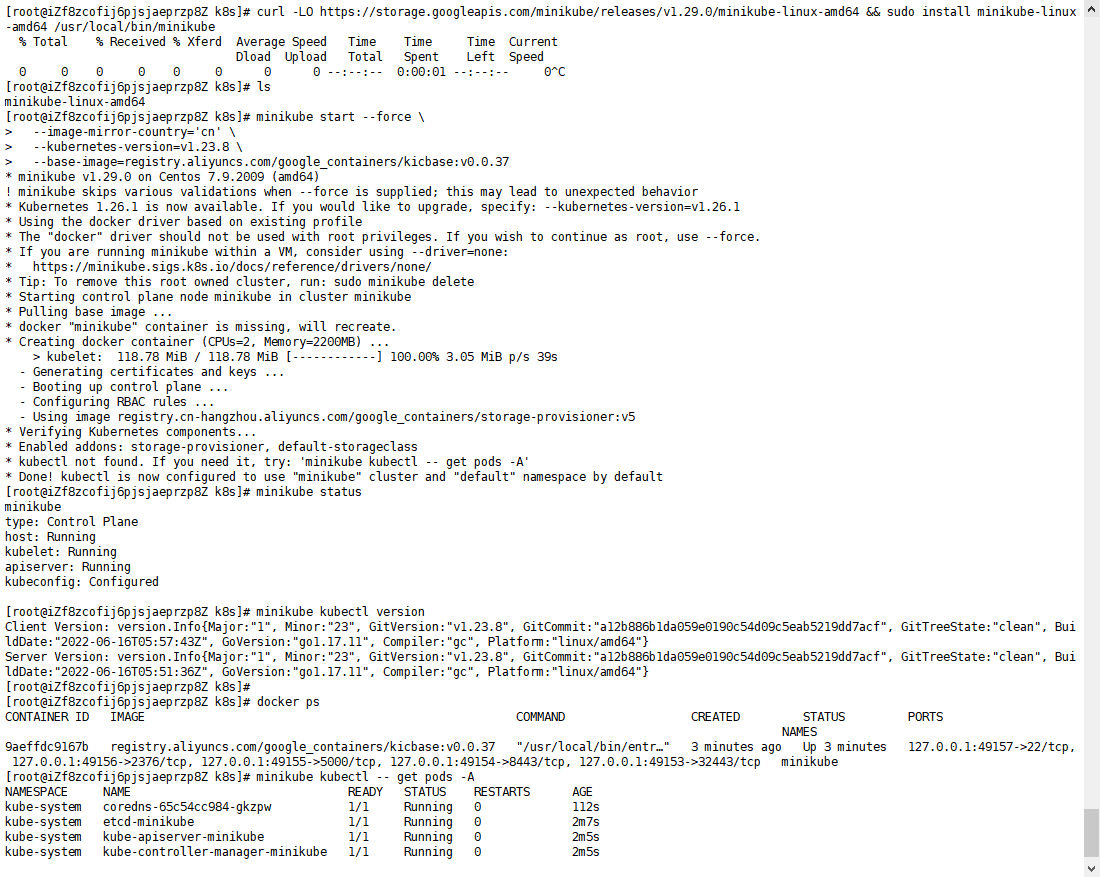
--registry-mirror="https://docker.mirrors.ustc.edu.cn" \

--registry-mirror="https://docker.1ms.run" \

--registry-mirror="https://docker.m.daocloud.io"

[root@iZf8zcofij6pjsjaeprzp8Z k8s]# minikube status

[root@iZf8zcofij6pjsjaeprzp8Z k8s]# minikube kubectl version

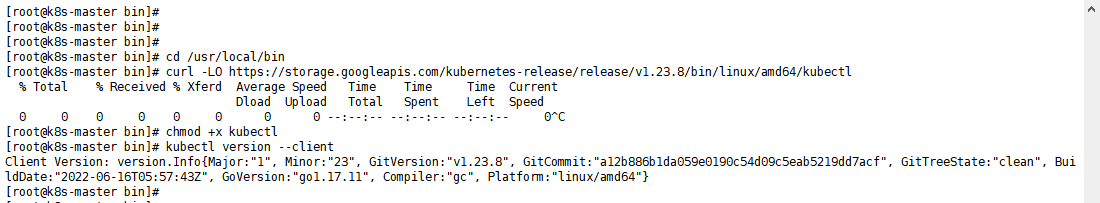


cd /usr/local/bin

curl -LO https://storage.googleapis.com/kubernetes-release/release/v1.23.8/bin/linux/amd64/kubectl

chmod +x kubectl

kubectl version --client



docker pull mysql:5.7

docker save mysql:5.7 -o mysql.tar

minikube image load mysql.tar

cat > mysql.yaml << EOF

apiVersion: v1

kind: Pod

metadata:

name: mysql

labels:

app: mysql

spec:

containers:

- name: mysql

image: mysql:5.7

imagePullPolicy: IfNotPresent

ports:

- containerPort: 3306

env:

- name: MYSQL\_ROOT\_PASSWORD

value: "Root@123"

volumeMounts:

- name: mysql-storage

mountPath: /var/lib/mysql

volumes:

- name: mysql-storage

hostPath:

path: /data/mysql

type: DirectoryOrCreate

---

apiVersion: v1

kind: Service

metadata:

name: mysql-service

spec:

type: NodePort

selector:

app: mysql

ports:

- port: 3306

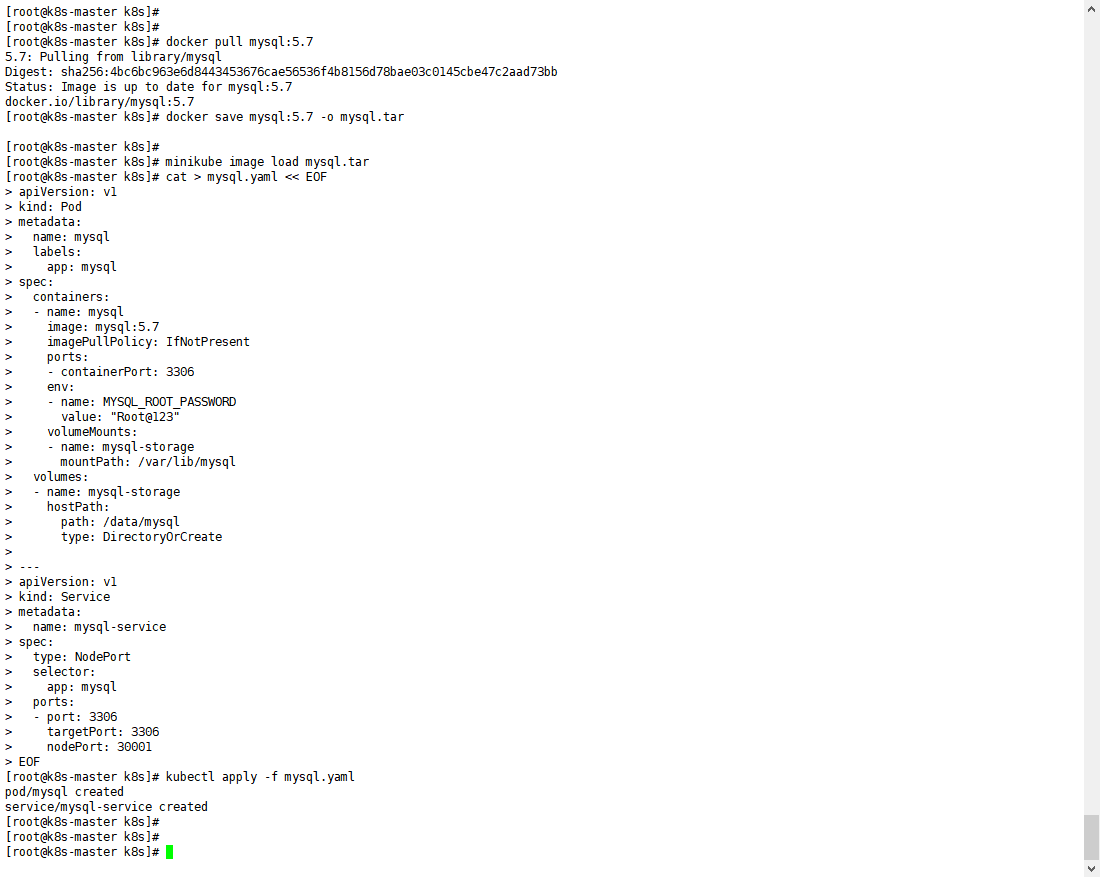
targetPort: 3306

nodePort: 30001

EOF

kubectl apply -f mysql.yaml

socat TCP4-LISTEN:30001,fork TCP4:192.168.49.2:30001 &



kubectl cp springboot1jxhb.sql mysql:/springboot1jxhb.sql

kubectl exec -it mysql -- bash

bash-4.2# mysql -uroot -p

Enter password:

mysql> create database springboot1jxhb;

Query OK, 1 row affected (0.00 sec)

mysql> exit

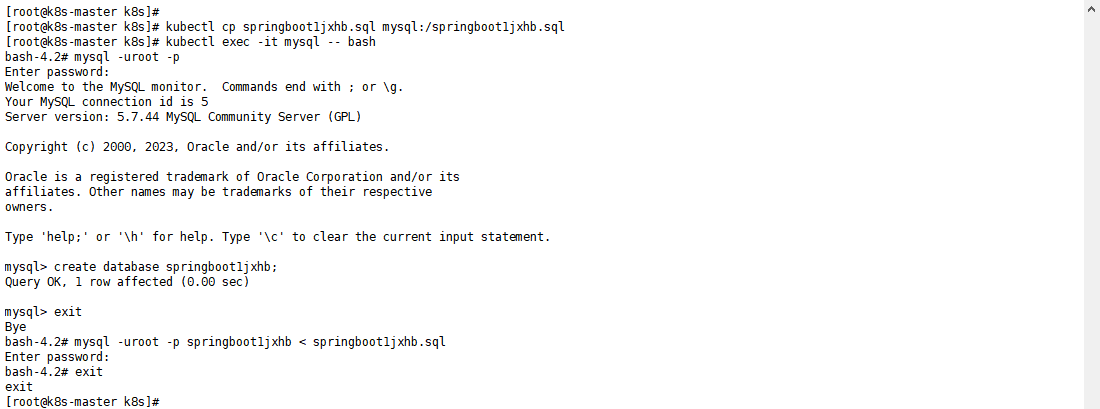
Bye

bash-4.2# mysql -uroot -p springboot1jxhb < springboot1jxhb.sql

Enter password:

bash-4.2# exit

exit



微服务改造application.yml添加：  
management:  
 endpoints:  
 web:  
 exposure:  
 include: prometheus,health,info  
 endpoint:  
 prometheus:  
 enabled: true  
 metrics:  
 export:  
 prometheus:  
 enabled: true

pom.xml添加：  
<dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-actuator</artifactId>  
</dependency>  
  
<!-- Prometheus 指标导出支持 -->  
<dependency>  
 <groupId>io.micrometer</groupId>  
 <artifactId>micrometer-registry-prometheus</artifactId>  
</dependency>

改造完成构建打包为springboot1jxhb-0.0.1-SNAPSHOT.jar

cat > Dockerfile.campus-convenience-service << EOF

FROM openjdk:17-jdk

MAINTAINER campus-convenience

RUN /bin/bash -c 'mkdir -p /data/campus-convenience'

VOLUME /data/campus-convenience

WORKDIR /data/campus-convenience

COPY ./static /data/campus-convenience/static

COPY ./springboot1jxhb-0.0.1-SNAPSHOT.jar /data/campus-convenience/springboot1jxhb-0.0.1-SNAPSHOT.jar

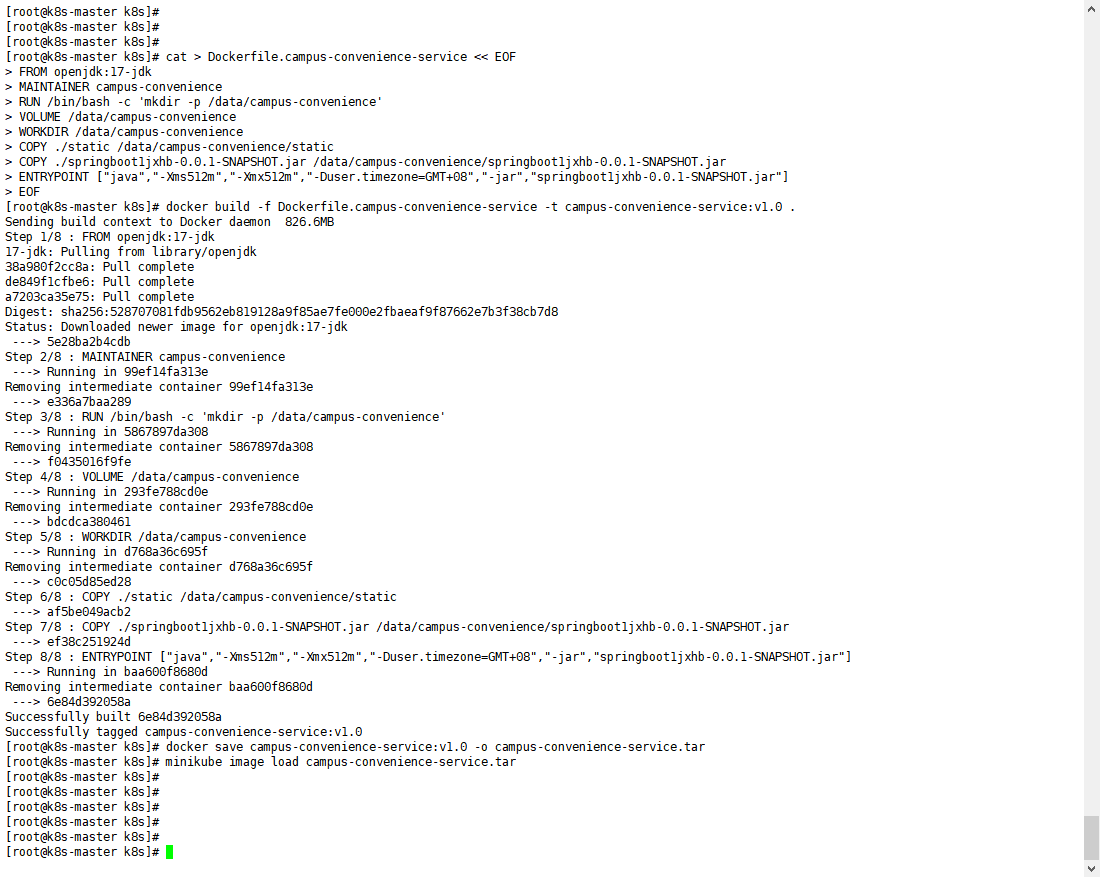
ENTRYPOINT ["java","-Xms512m","-Xmx512m","-Duser.timezone=GMT+08","-jar","springboot1jxhb-0.0.1-SNAPSHOT.jar"]

EOF

docker build -f Dockerfile.campus-convenience-service -t campus-convenience-service:v1.0 .

docker save campus-convenience-service:v1.0 -o campus-convenience-service.tar

minikube image load campus-convenience-service.tar



cat > campus-convenience.yaml << EOF

apiVersion: apps/v1

kind: Deployment

metadata:

name: campus-convenience

spec:

replicas: 2

selector:

matchLabels:

app: campus-convenience

template:

metadata:

labels:

app: campus-convenience

spec:

containers:

- name: campus-convenience

image: campus-convenience-service:v1.0

imagePullPolicy: IfNotPresent

ports:

- containerPort: 8080

---

apiVersion: v1

kind: Service

metadata:

name: campus-convenience-service

spec:

selector:

app: campus-convenience

ports:

- name: campus-convenience

port: 8080

targetPort: 8080

nodePort: 31000

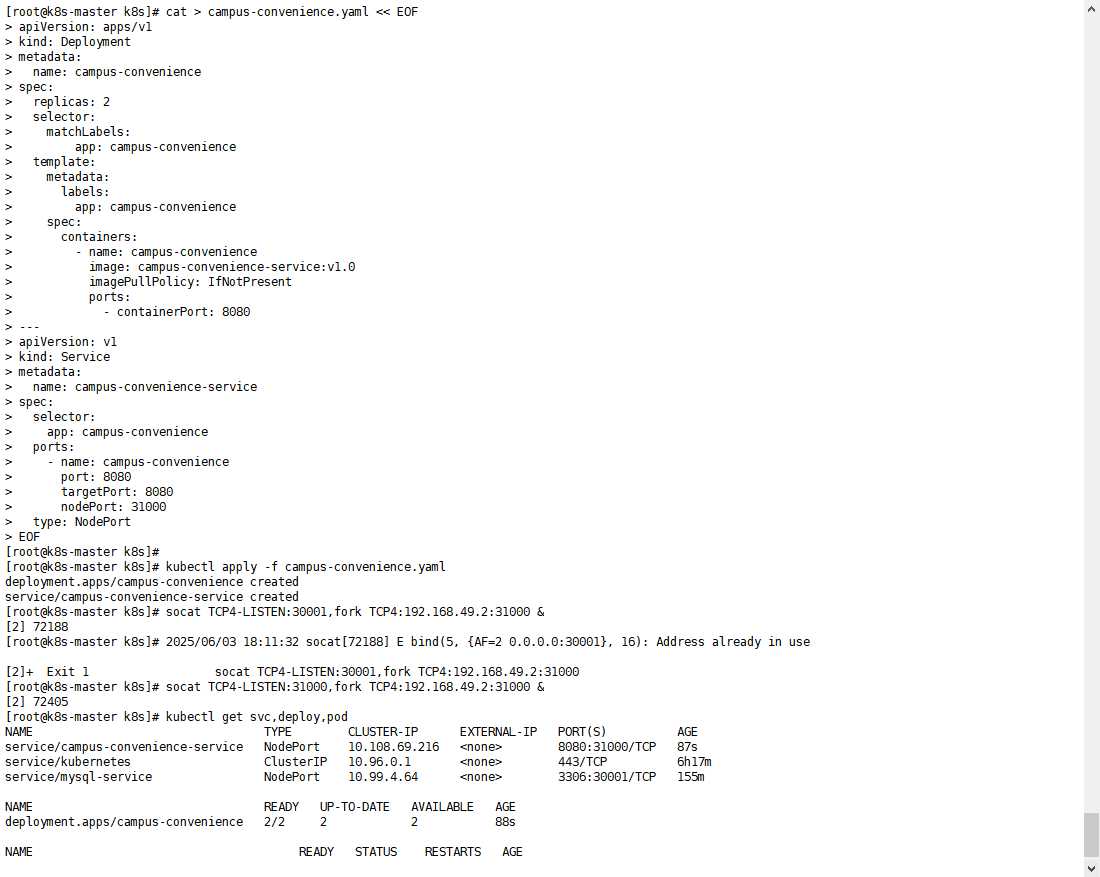
type: NodePort

EOF

kubectl apply -f campus-convenience.yaml

socat TCP4-LISTEN:31000,fork TCP4:192.168.49.2:31000 &

kubectl get svc,deploy,pod





访问指标：http://47.113.217.50:31000/springboot1jxhb/actuator/prometheus

http://47.113.217.50:31000/springboot1jxhb/admin/dist/index.html#/login

账号admin/admin

http://47.113.217.50:31000/springboot1jxhb/front/dist/index.html

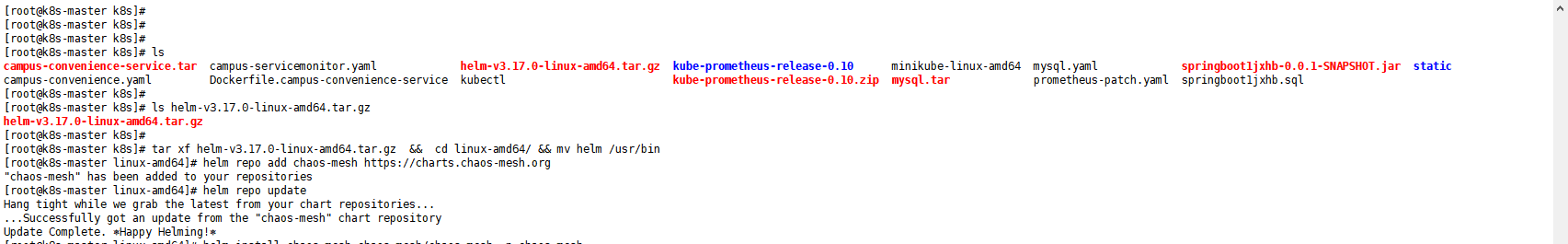
账号1/123456

## ChaosMesh

tar xf helm-v3.17.0-linux-amd64.tar.gz && cd linux-amd64/ && mv helm /usr/bin

helm repo add chaos-mesh https://charts.chaos-mesh.org

helm repo update



docker pull ghcr.io/chaos-mesh/chaos-daemon:v2.7.2 && docker pull ghcr.io/chaos-mesh/chaos-dashboard:v2.7.2 && docker pull ghcr.io/chaos-mesh/chaos-coredns:v0.2.6

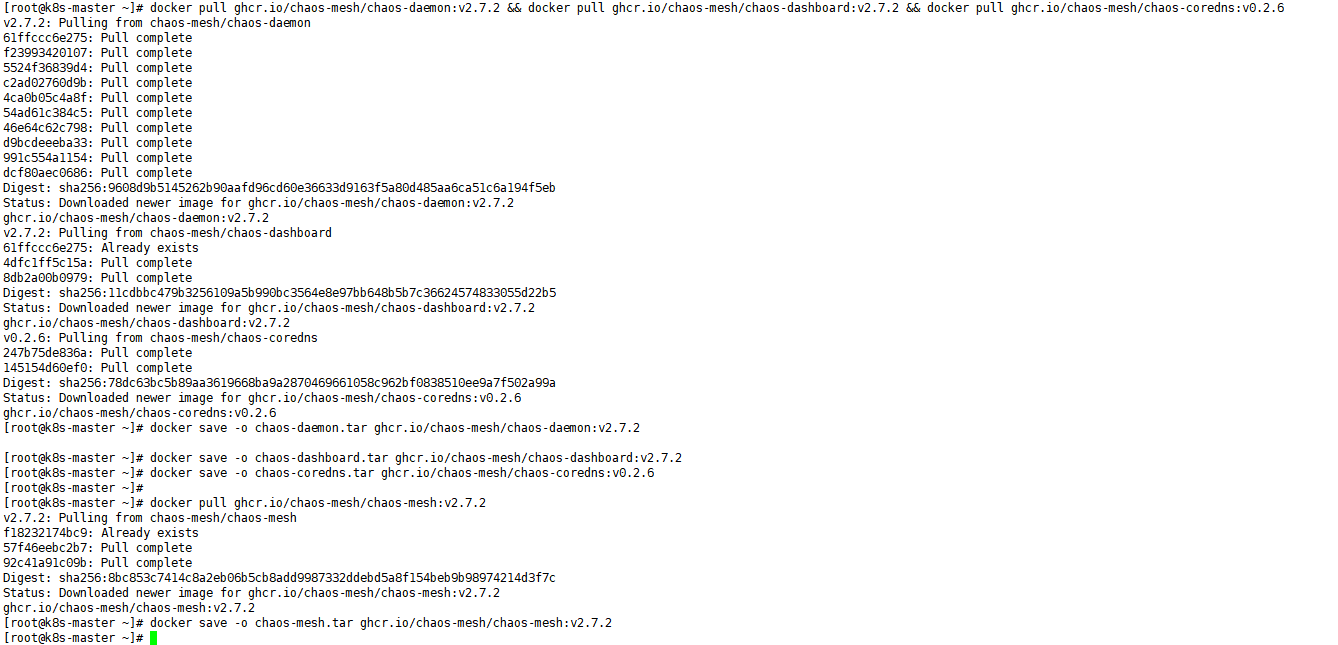
docker save -o chaos-daemon.tar ghcr.io/chaos-mesh/chaos-daemon:v2.7.2

docker save -o chaos-dashboard.tar ghcr.io/chaos-mesh/chaos-dashboard:v2.7.2

docker save -o chaos-coredns.tar ghcr.io/chaos-mesh/chaos-coredns:v0.2.6

docker pull ghcr.io/chaos-mesh/chaos-mesh:v2.7.2

docker save -o chaos-mesh.tar ghcr.io/chaos-mesh/chaos-mesh:v2.7.2



minikube image load chaos-daemon.tar

minikube image load chaos-dashboard.tar

minikube image load chaos-coredns.tar

minikube image load chaos-mesh.tar



helm upgrade --install chaos-mesh chaos-mesh/chaos-mesh -n chaos-mesh \

--create-namespace \

--set chaosDaemon.bpf.enabled=false \

--set chaosDaemon.runtime=docker \

--set chaosDaemon.env.DOCKER\_API\_VERSION=1.40 \

--set chaosDaemon.image.registry="" \

--set chaosDaemon.image.repository=chaos-mesh/chaos-daemon \

--set chaosDaemon.image.tag=v2.7.2 \

--set dashboard.create=true \

--set dashboard.image.registry="" \

--set dashboard.image.repository=chaos-mesh/chaos-dashboard \

--set dashboard.image.tag=v2.7.2 \

--set dnsServer.image.registry="" \

--set dnsServer.image.repository=chaos-mesh/chaos-coredns \

--set dnsServer.image.tag=v0.2.6 \

--set controllerManager.image.registry="" \

--set controllerManager.image.repository=chaos-mesh/chaos-mesh \

--set controllerManager.image.tag=v2.7.2 \

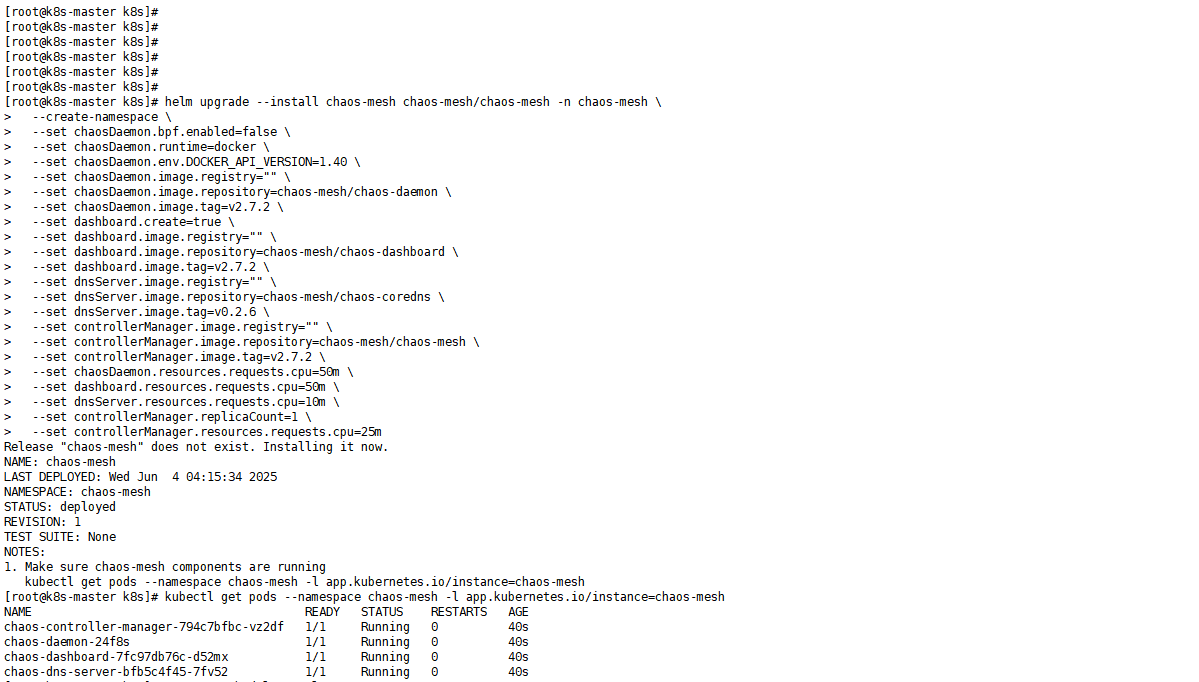
--set chaosDaemon.resources.requests.cpu=50m \

--set dashboard.resources.requests.cpu=50m \

--set dnsServer.resources.requests.cpu=10m \

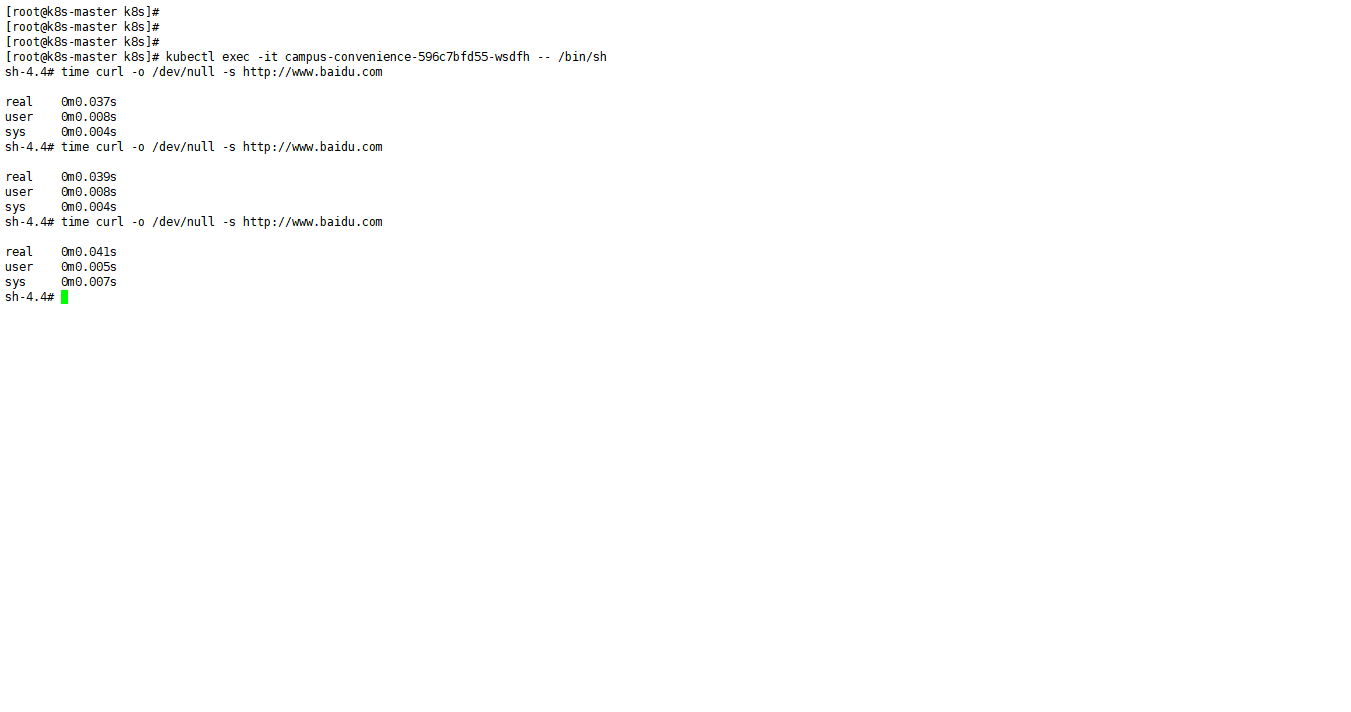
--set controllerManager.replicaCount=1 \

--set controllerManager.resources.requests.cpu=25m



kubectl exec -it campus-convenience-596c7bfd55-wsdfh -- /bin/sh

time curl -o /dev/null -s http://www.baidu.com



cat > network-delay.yaml << EOF

apiVersion: chaos-mesh.org/v1alpha1

kind: NetworkChaos

metadata:

name: network-delay-example

spec:

action: delay

mode: all

selector:

labelSelectors:

"app": "campus-convenience"

delay:

latency: "100ms"

duration: "5m"

EOF

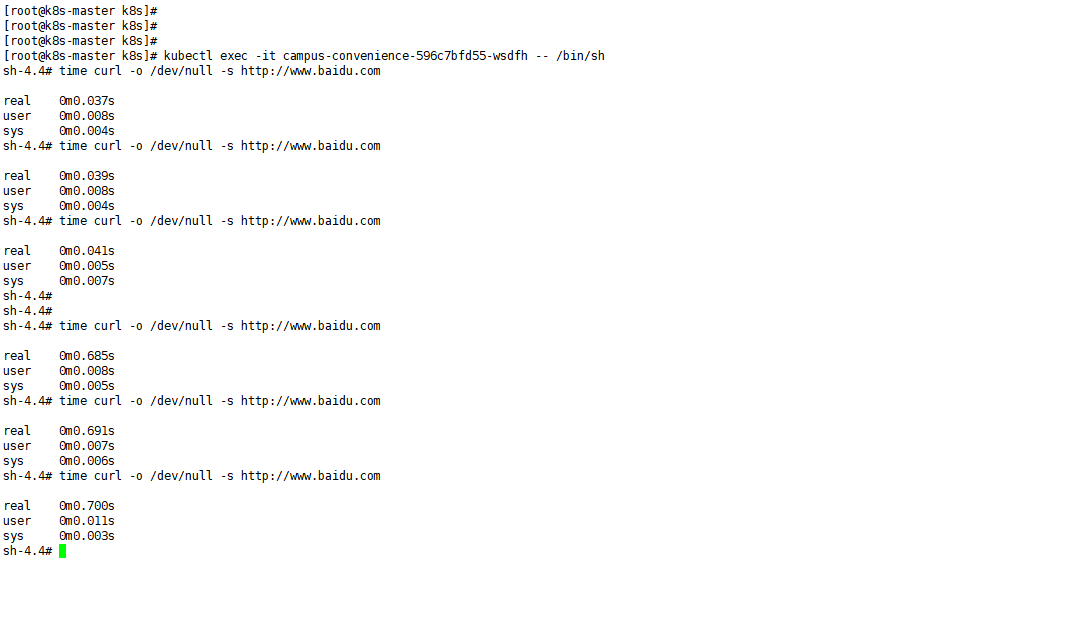
kubectl apply -f network-delay.yaml

kubectl get NetworkChaos



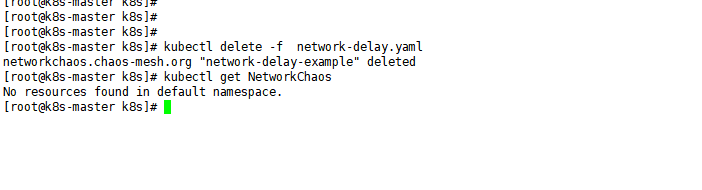
再次执行

time curl -o /dev/null -s http://www.baidu.com



删除实验影响

kubectl delete -f network-delay.yaml

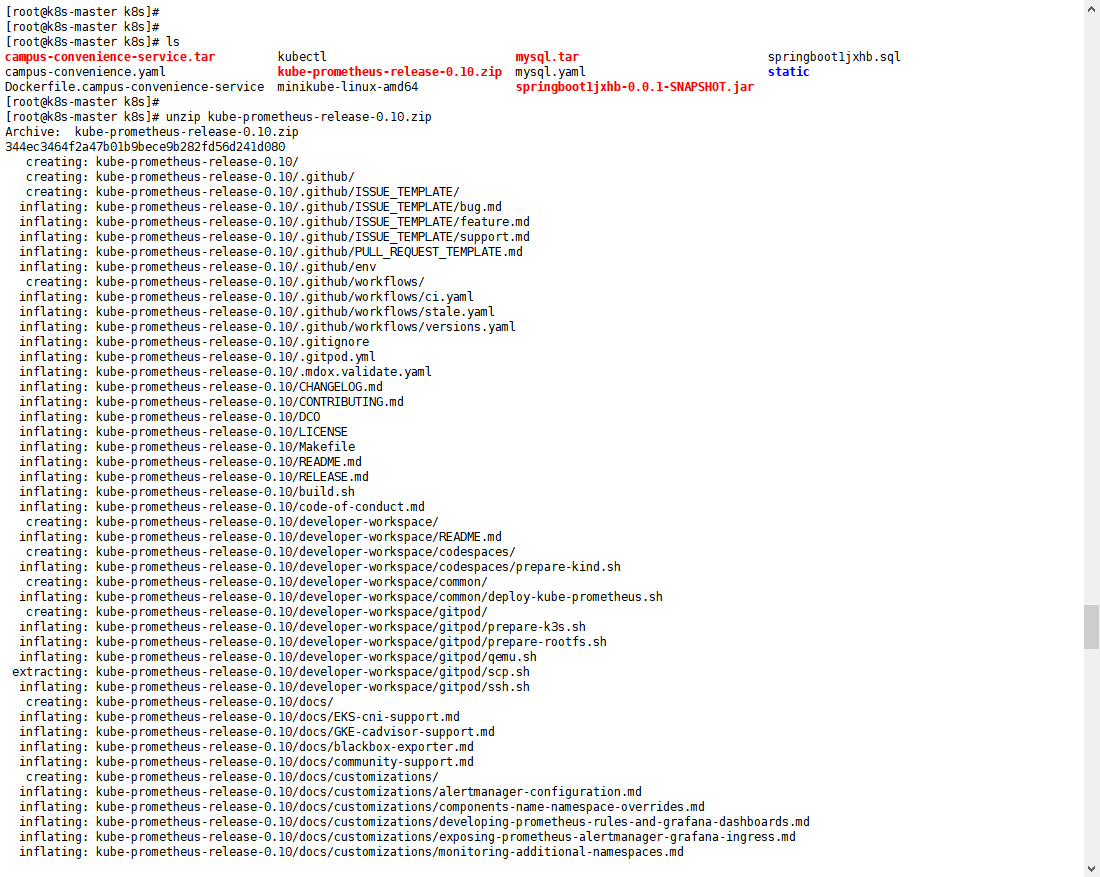


## Prometheus & Grafana

https://github.com/prometheus-operator/kube-prometheus/tree/release-0.10

下载zip包上传至服务器

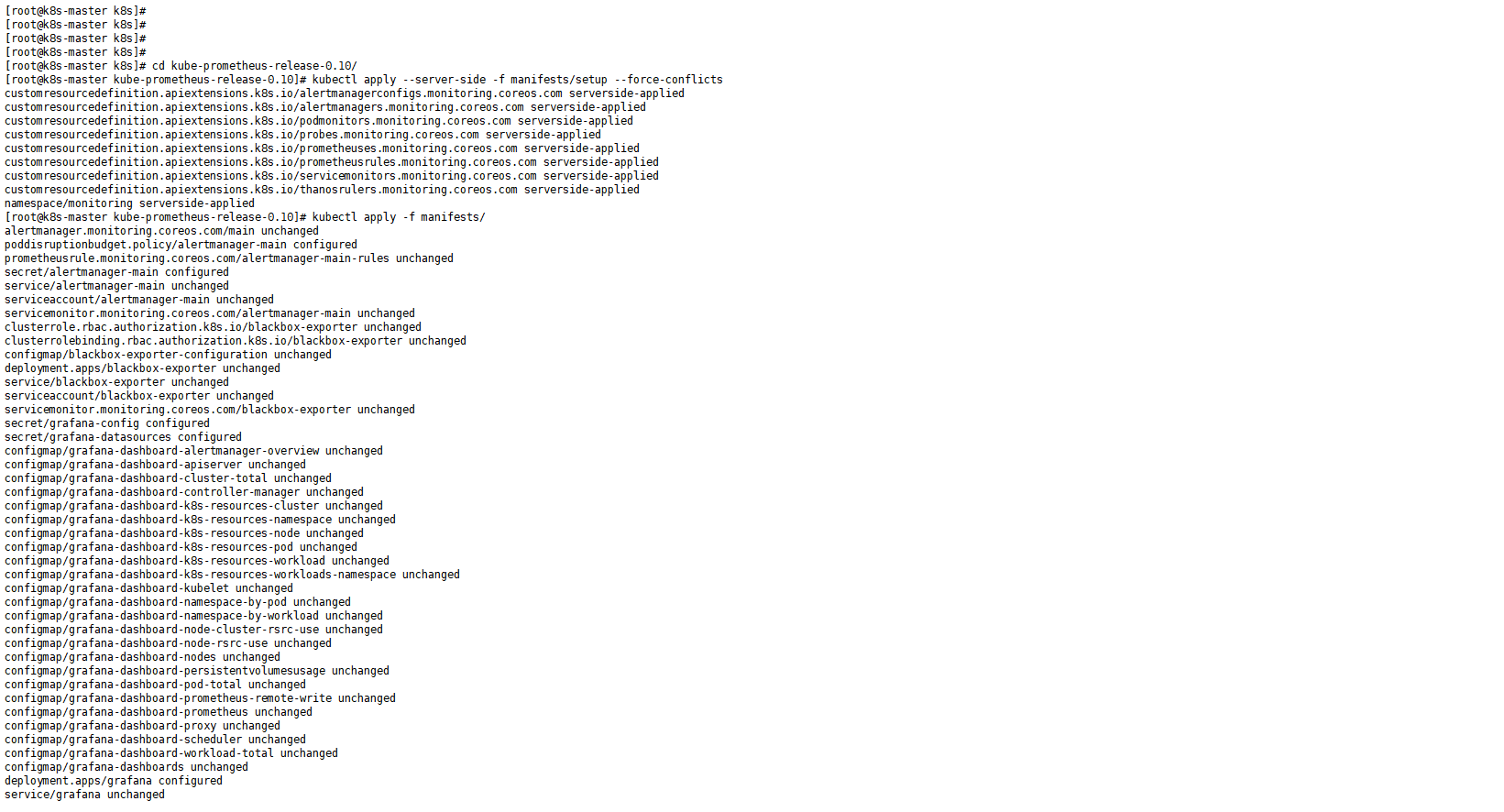
unzip kube-prometheus-release-0.10.zip



调整部分国内镜像后执行

kubectl apply --server-side -f manifests/setup --force-conflicts

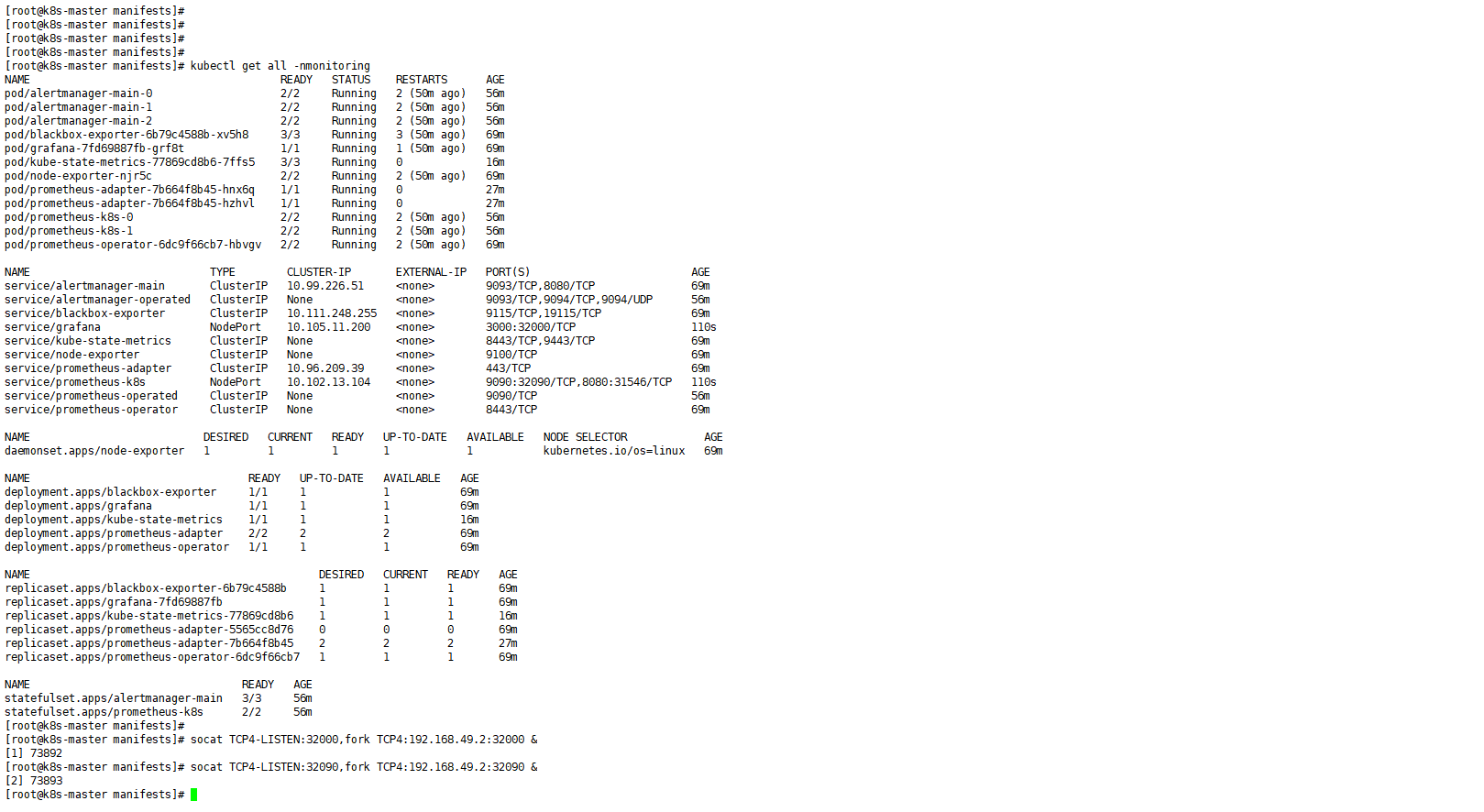
kubectl apply -f manifests/



kubectl get all -nmonitoring

socat TCP4-LISTEN:32000,fork TCP4:192.168.49.2:32000 &

socat TCP4-LISTEN:32090,fork TCP4:192.168.49.2:32090 &



prometheus：

http://47.113.217.50:32090

grafana

http://47.113.217.50:32000/login

账号admin/admin

cat > campus-servicemonitor.yaml << EOF

apiVersion: v1

kind: Service

metadata:

name: campus-convenience-metrics

namespace: default

labels:

app: springboot-metrics

spec:

selector:

app: campus-convenience

ports:

- name: metrics

port: 8080

targetPort: 8080

---

apiVersion: monitoring.coreos.com/v1

kind: ServiceMonitor

metadata:

name: campus-convenience-monitor

namespace: default

labels:

release: prometheus

app: springboot-metrics

spec:

selector:

matchLabels:

app: springboot-metrics

endpoints:

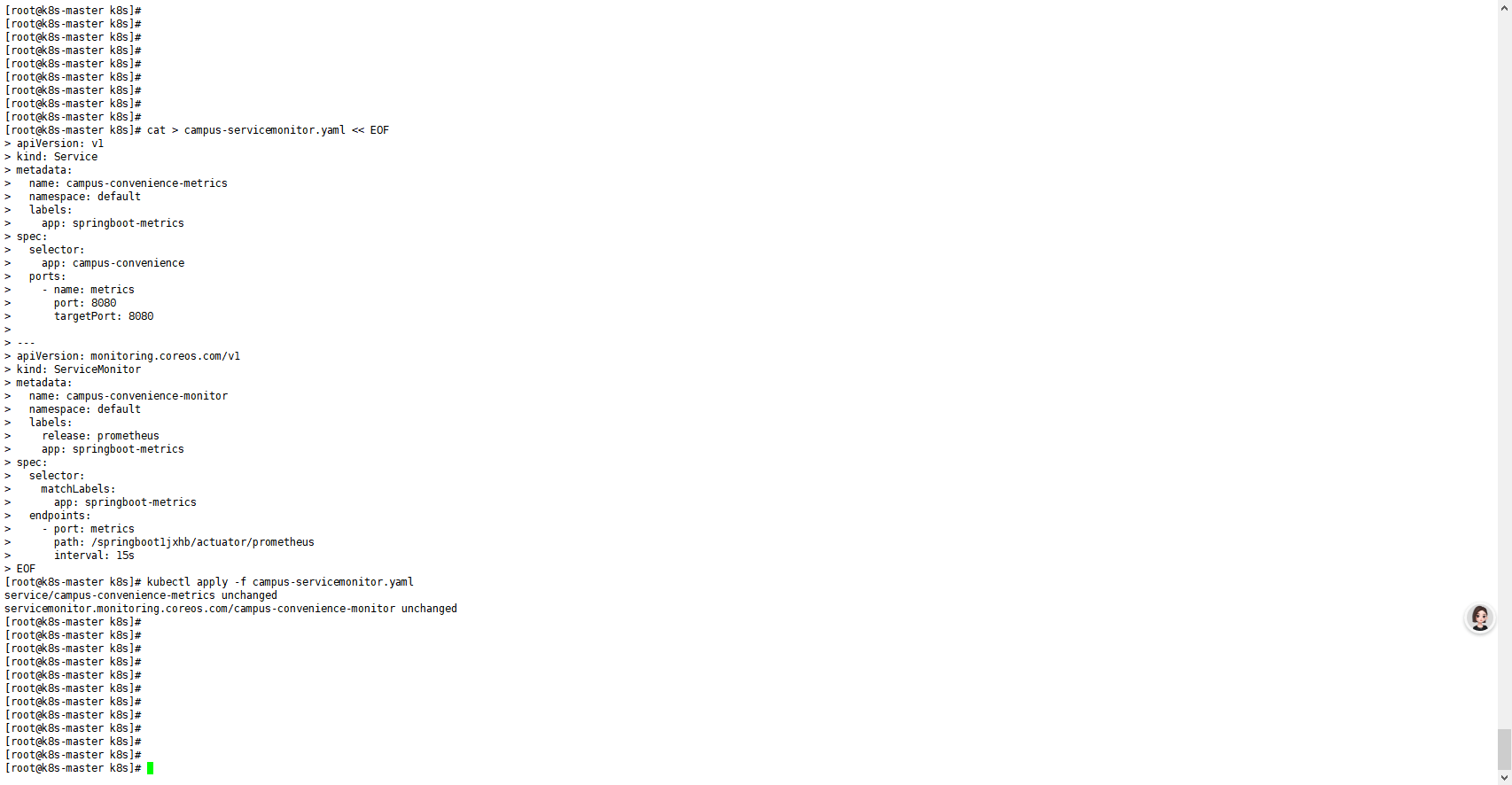
- port: metrics

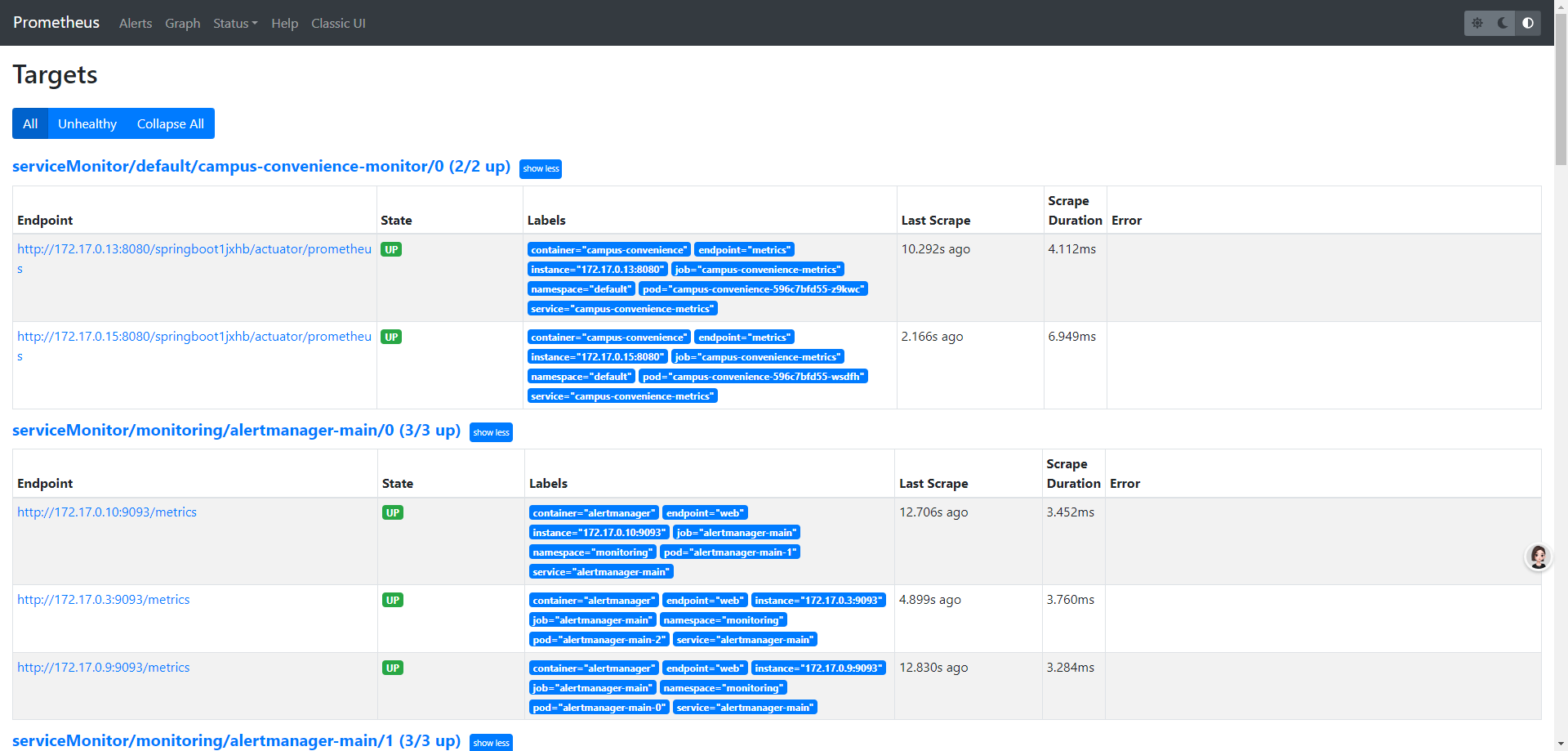
path: /springboot1jxhb/actuator/prometheus

interval: 15s

EOF

kubectl apply -f campus-servicemonitor.yaml





响应时间（单个pod）

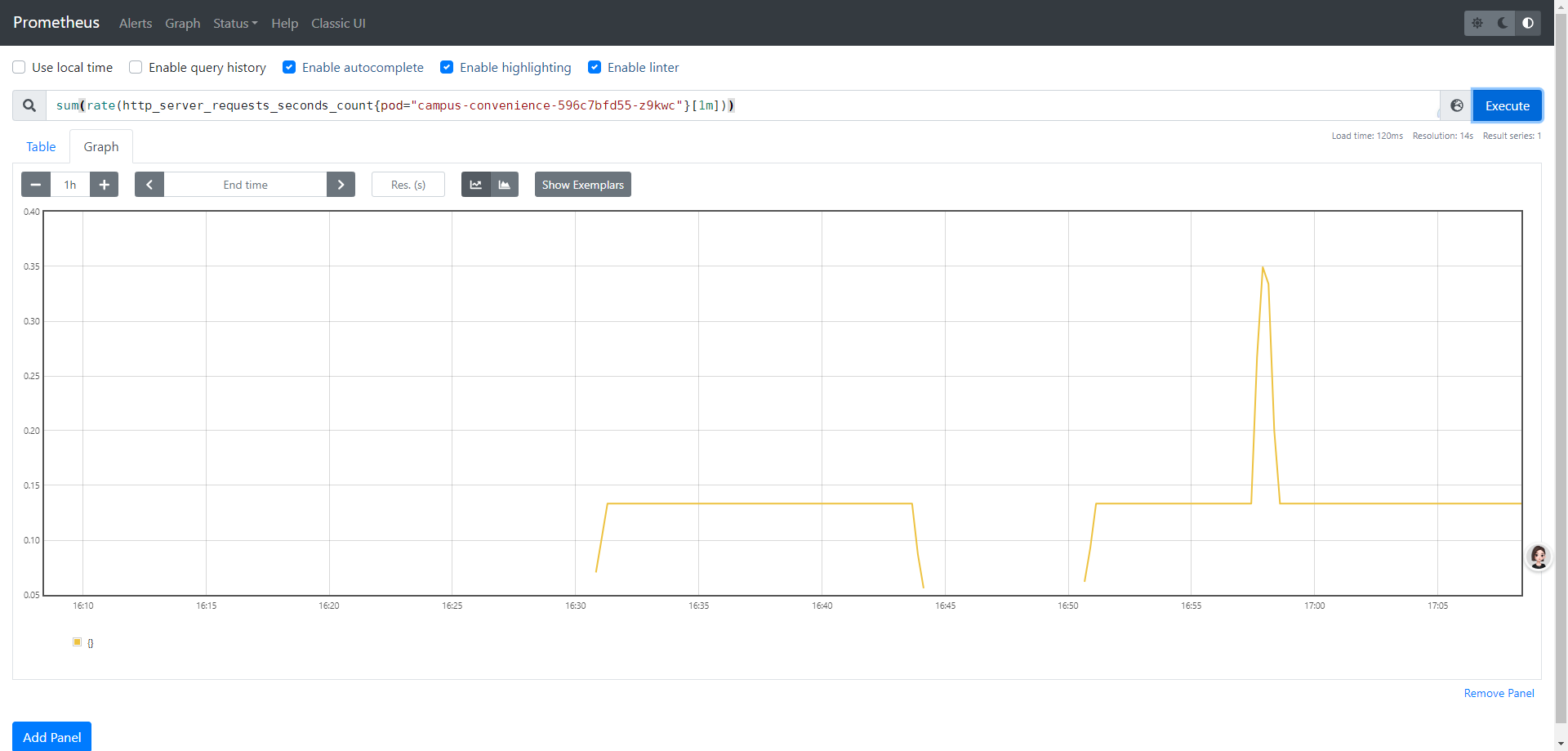
sum(rate(http\_server\_requests\_seconds\_count{pod="campus-convenience-596c7bfd55-z9kwc"}[1m]))

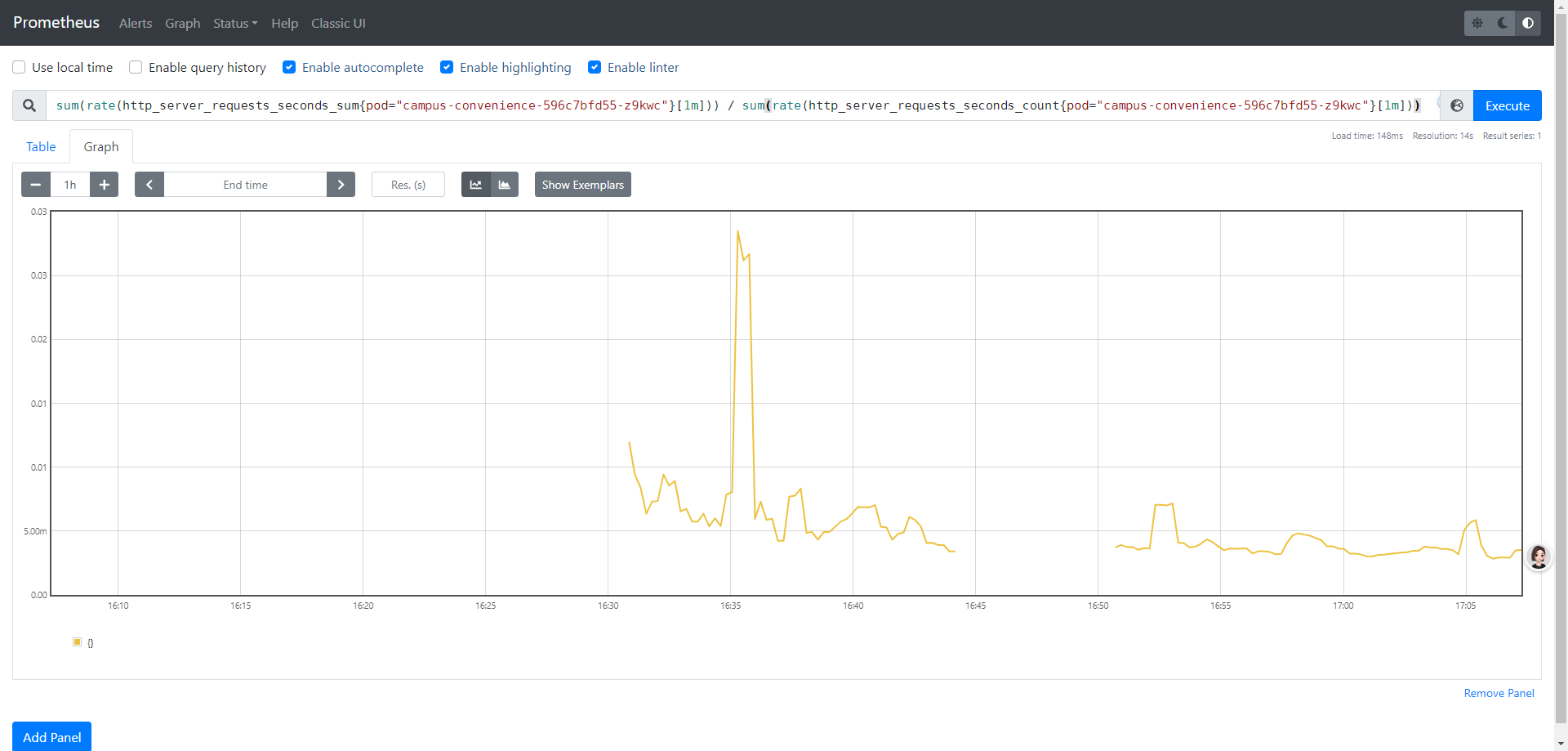
过去 1 分钟内的请求平均响应时间（单个pod）

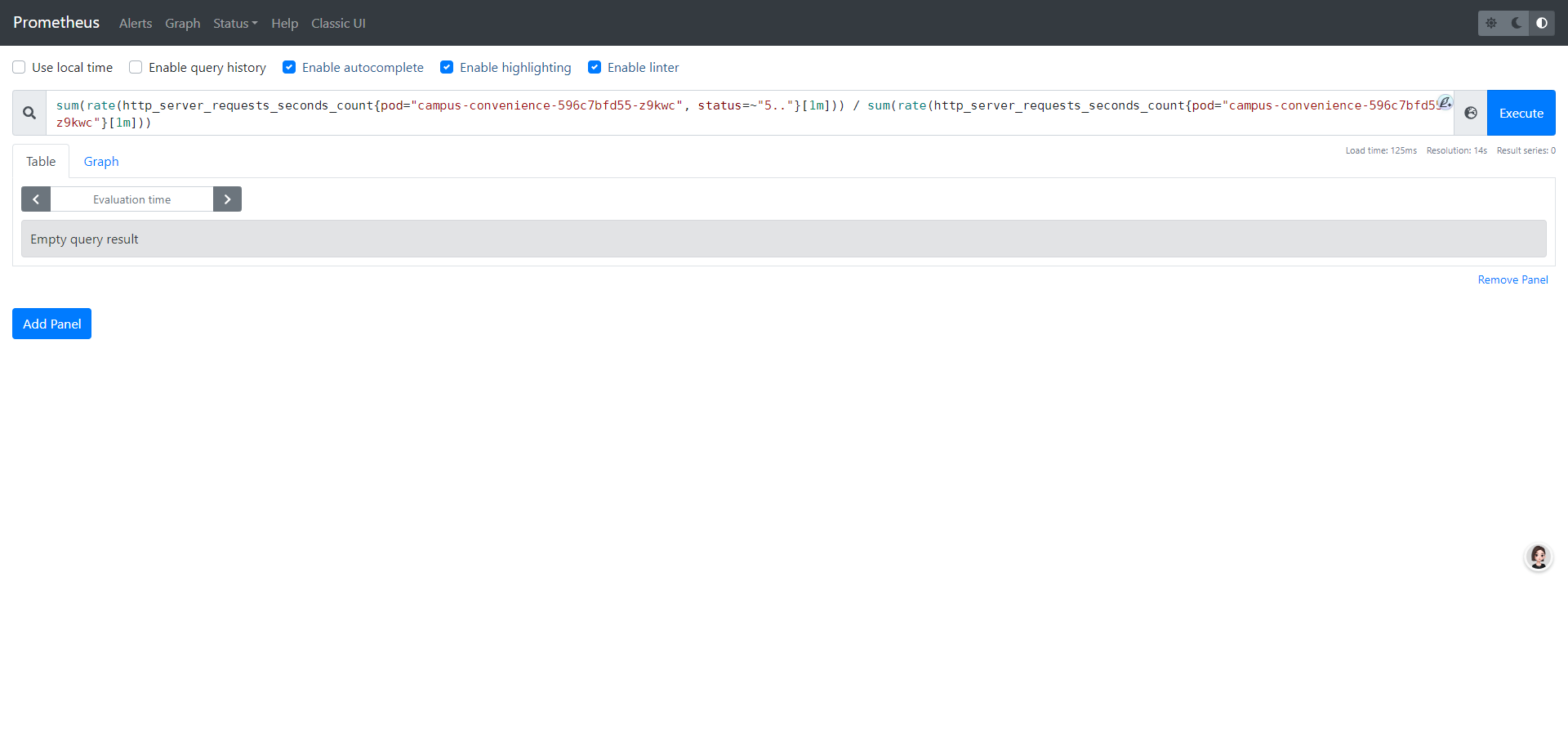
sum(rate(http\_server\_requests\_seconds\_sum{pod="campus-convenience-596c7bfd55-z9kwc"}[1m])) / sum(rate(http\_server\_requests\_seconds\_count{pod="campus-convenience-596c7bfd55-z9kwc"}[1m]))

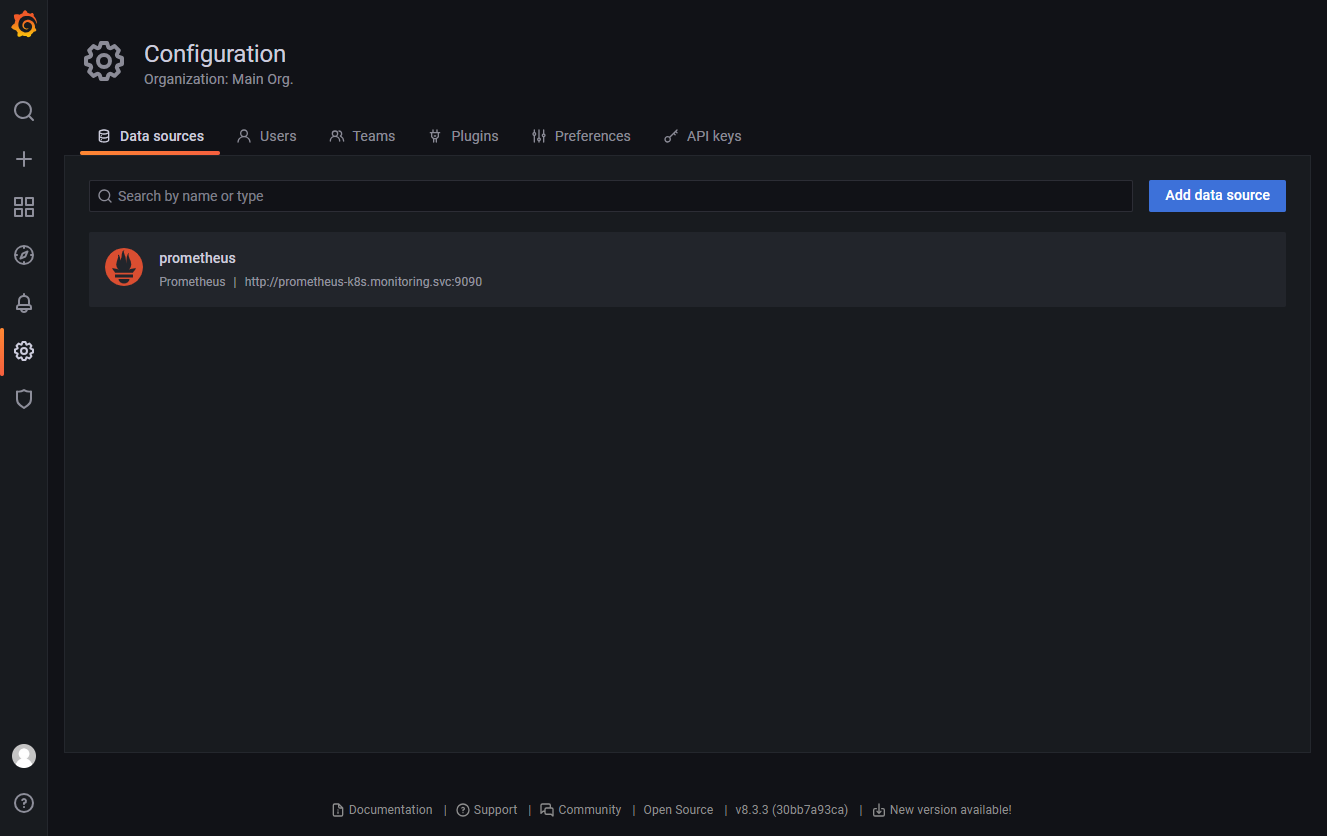
错误率（单个pod）

sum(rate(http\_server\_requests\_seconds\_count{pod="campus-convenience-596c7bfd55-z9kwc", status=~"5.."}[1m])) / sum(rate(http\_server\_requests\_seconds\_count{pod="campus-convenience-596c7bfd55-z9kwc"}[1m]))

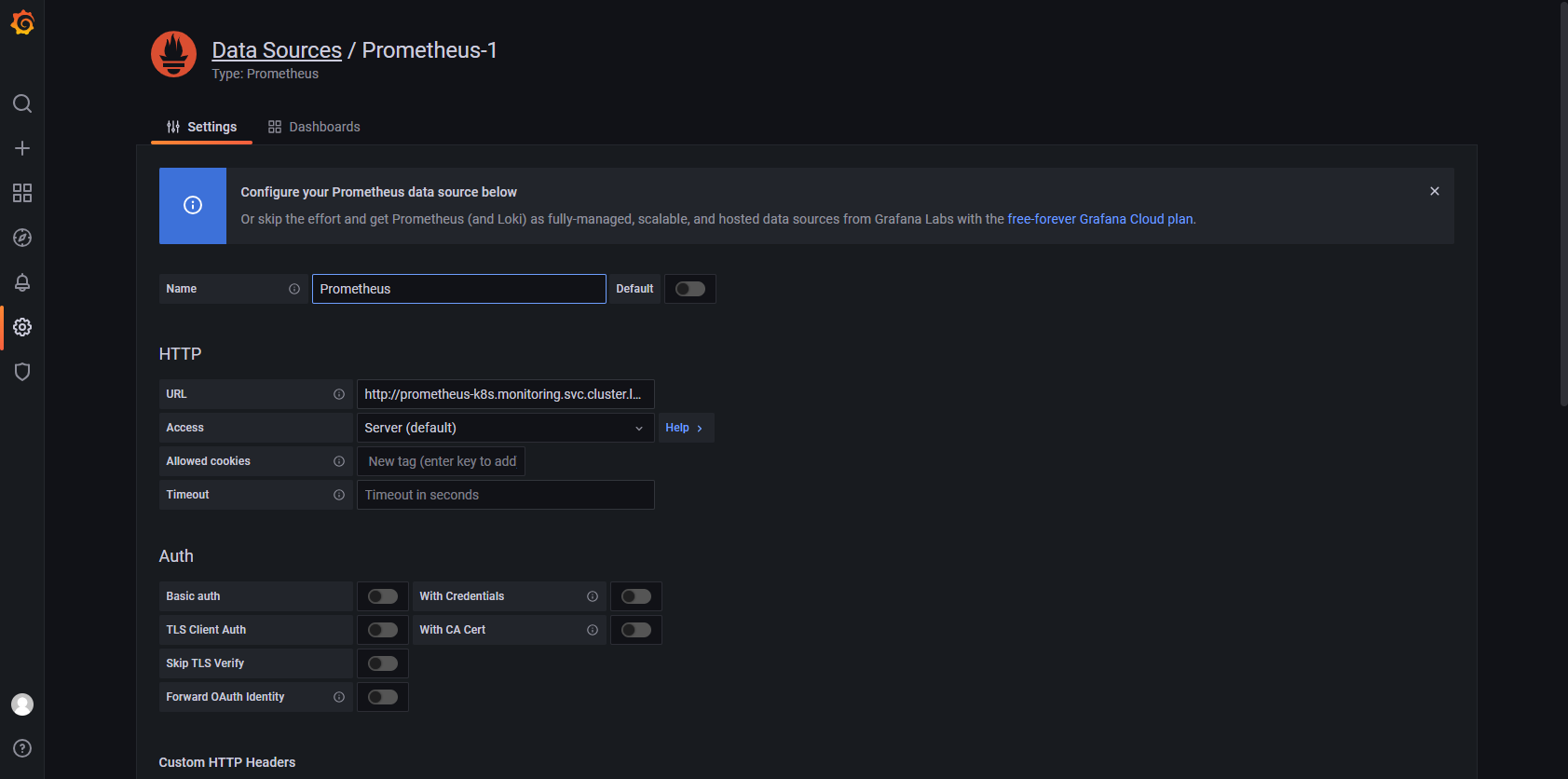


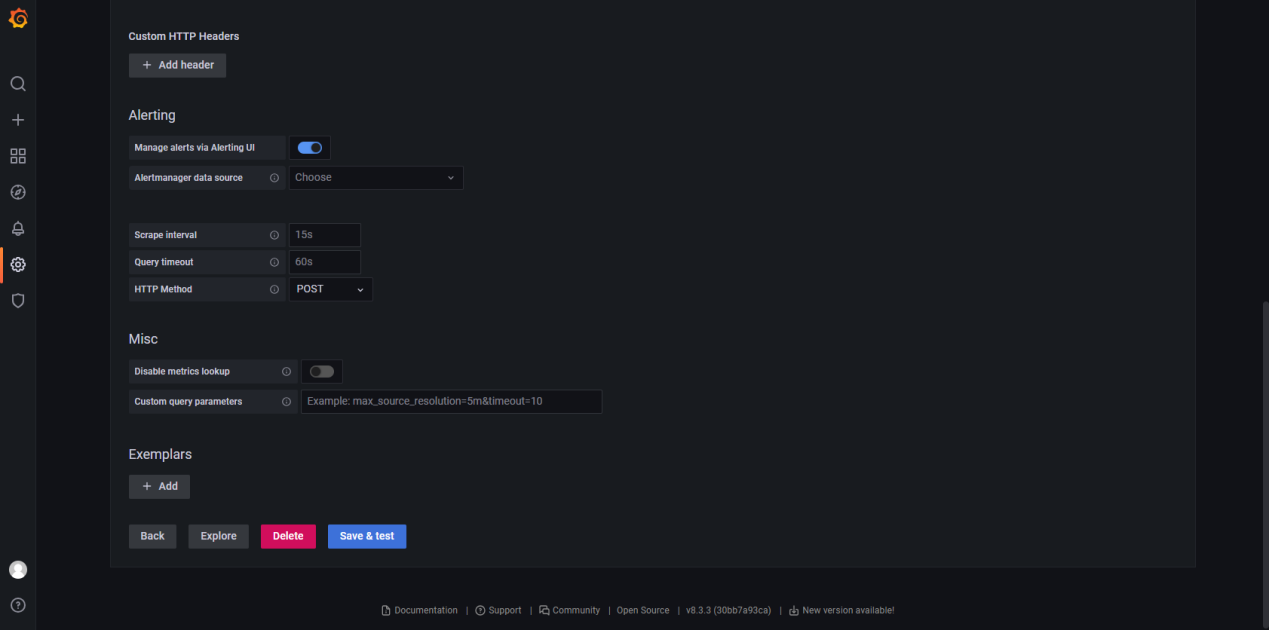


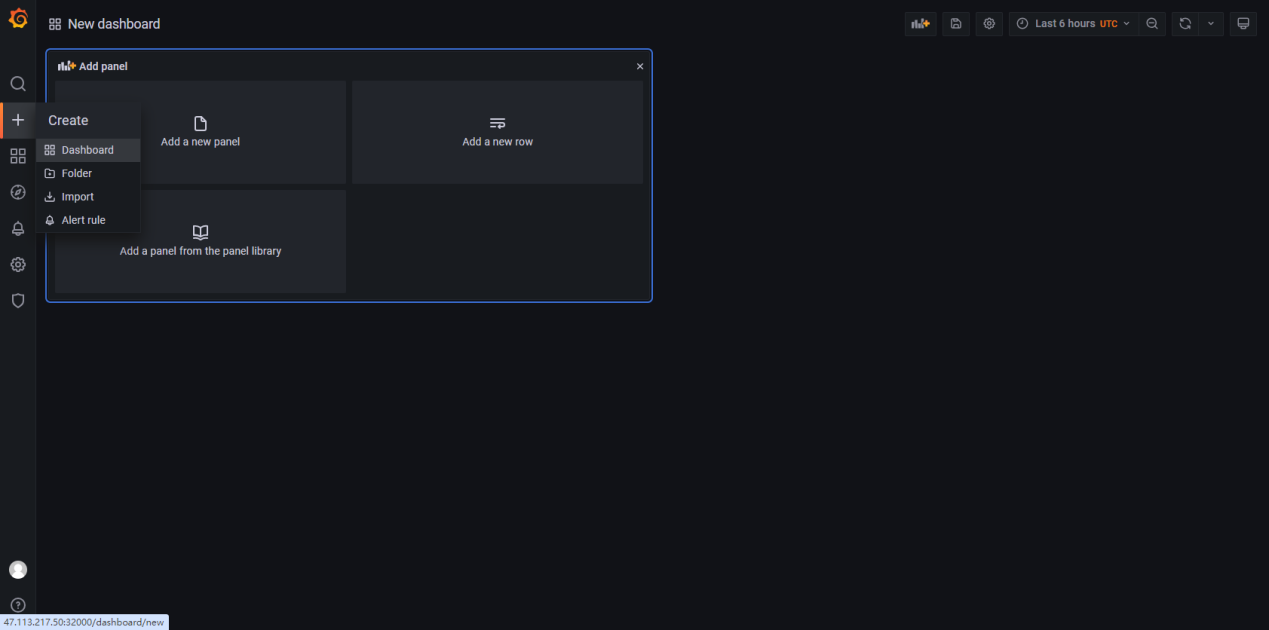


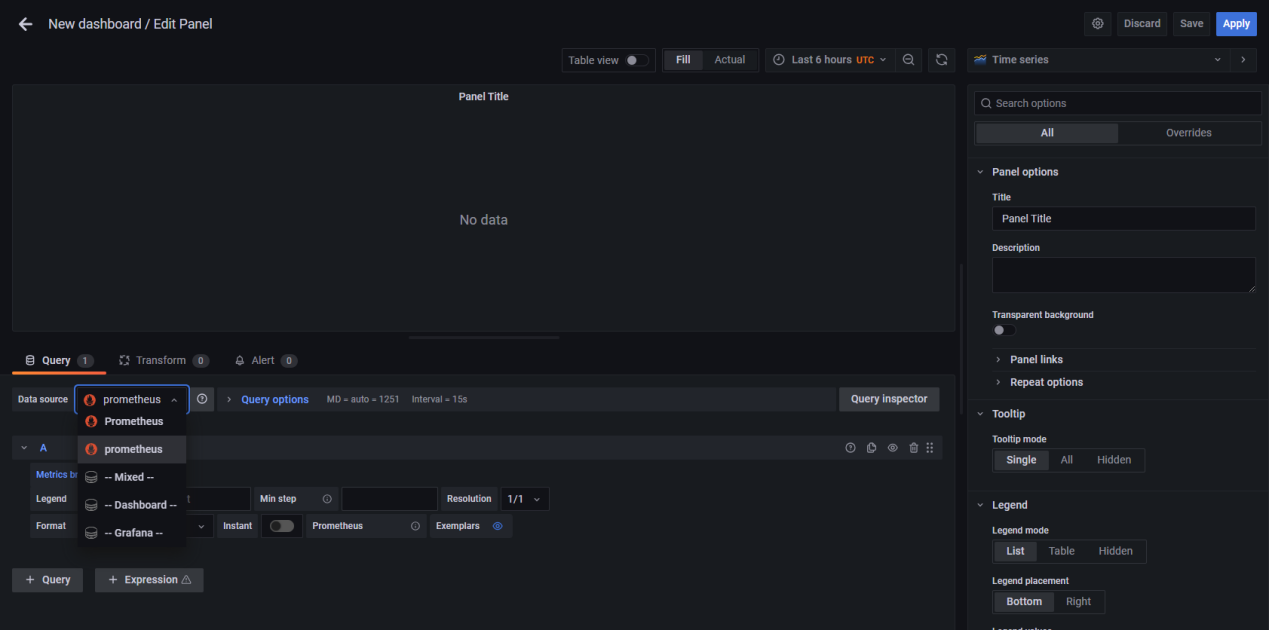


下面地址填写：http://prometheus-k8s.monitoring.svc.cluster.local:9090



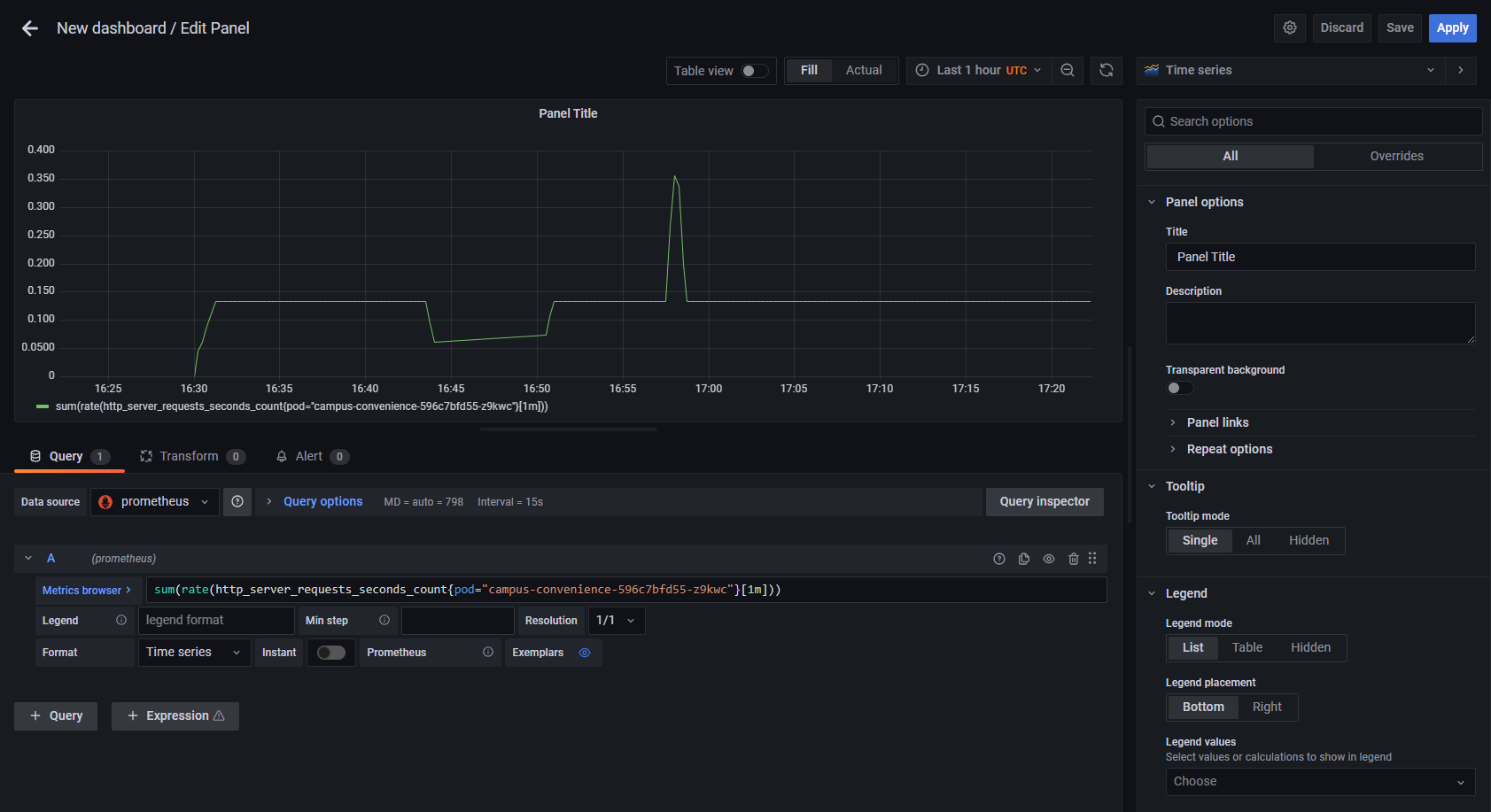






响应时间（单个pod）

sum(rate(http\_server\_requests\_seconds\_count{pod="campus-convenience-596c7bfd55-z9kwc"}[1m]))



过去 1 分钟内的请求平均响应时间（单个pod）

sum(rate(http\_server\_requests\_seconds\_sum{pod="campus-convenience-596c7bfd55-z9kwc"}[1m])) / sum(rate(http\_server\_requests\_seconds\_count{pod="campus-convenience-596c7bfd55-z9kwc"}[1m]))

