腾讯云

IT大咖说 知识分享平台

基于腾讯云的kubernetes实践

包健@腾讯云

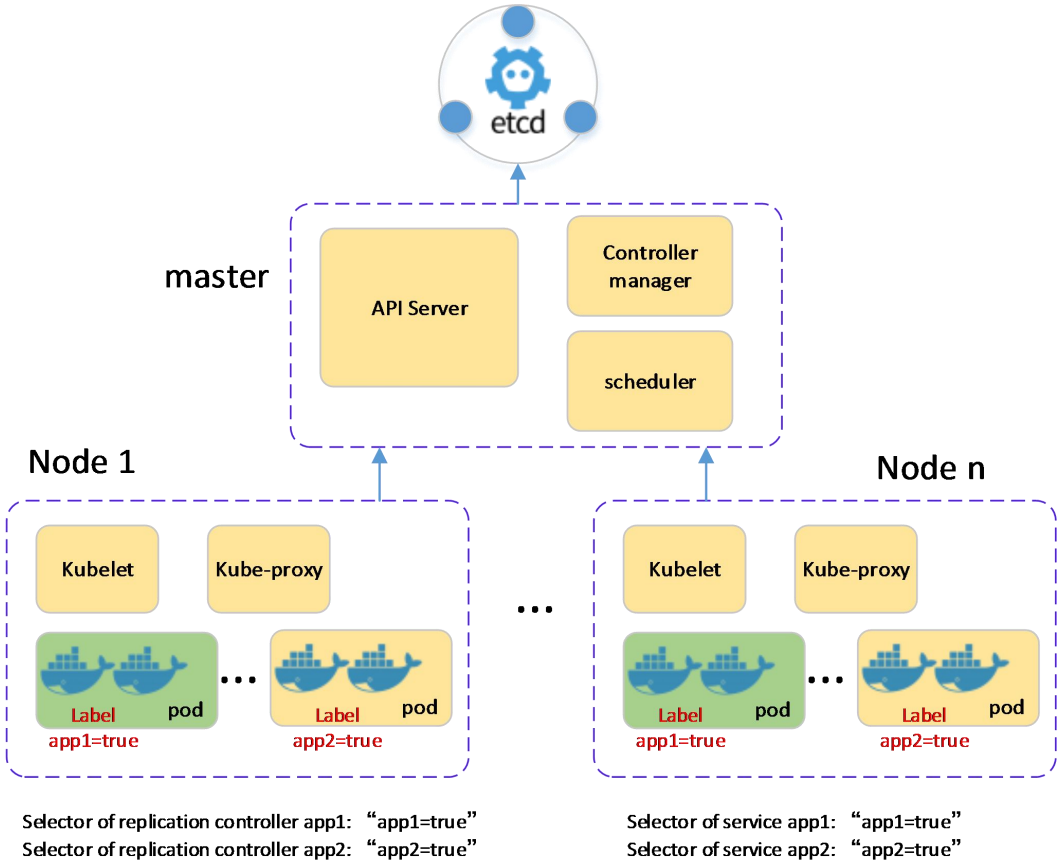
为什么选择K8S

三大开源容器编排工具在 Github 上的活跃数据(2016.9 月)			
技术平台	社区关注数	社区喜爱数	社区参与数
Kubernetes	1461	17416	5647
Swarm	339	4033	814
Mesos	53	387	72
Marathon (Mesos 依赖框架)	353	2187	630

- 开源、社区活跃度最高
- 对公有云的支持完善，像volume，loadblance、router等都有可以通过plugin来提供
- 为分布式而生，相关概念完备，如服务发现、健康检查、ingress、命名空间等
- 使用人数多，70%以上的企业客户在使用 kubernetes 来搭建容器服务

相比之下，Docker swarm 的背后厂商以 Docker 公司自己为主，而 Mesos 则以 Apache 开源基金会为主。

k8s简介



Kubernetes常用名词	说明
pod	容器组，Kubernetes的基本资源
replication controller	Pod的副本控制器
service	Pod的流量访问控制器
label	标签
node	集群节点，运行pod
master	集群master，负责资源的管理、调度
etcd	集群资源持久化db

01

控制台

a、集群管理：用户通过控制台管理集群，不需手动配置搭建

b、封装服务、配置项、pod等概念封装到控制台

c、提供服务事件、容器日志和webshell等工具

02

对接腾讯云IAAS层

a、基于VPC的容器网络，可和外部CDB等资源打通

b、支持cbs盘的volume和storage class

c、基于腾讯云LB的ingress controller

03

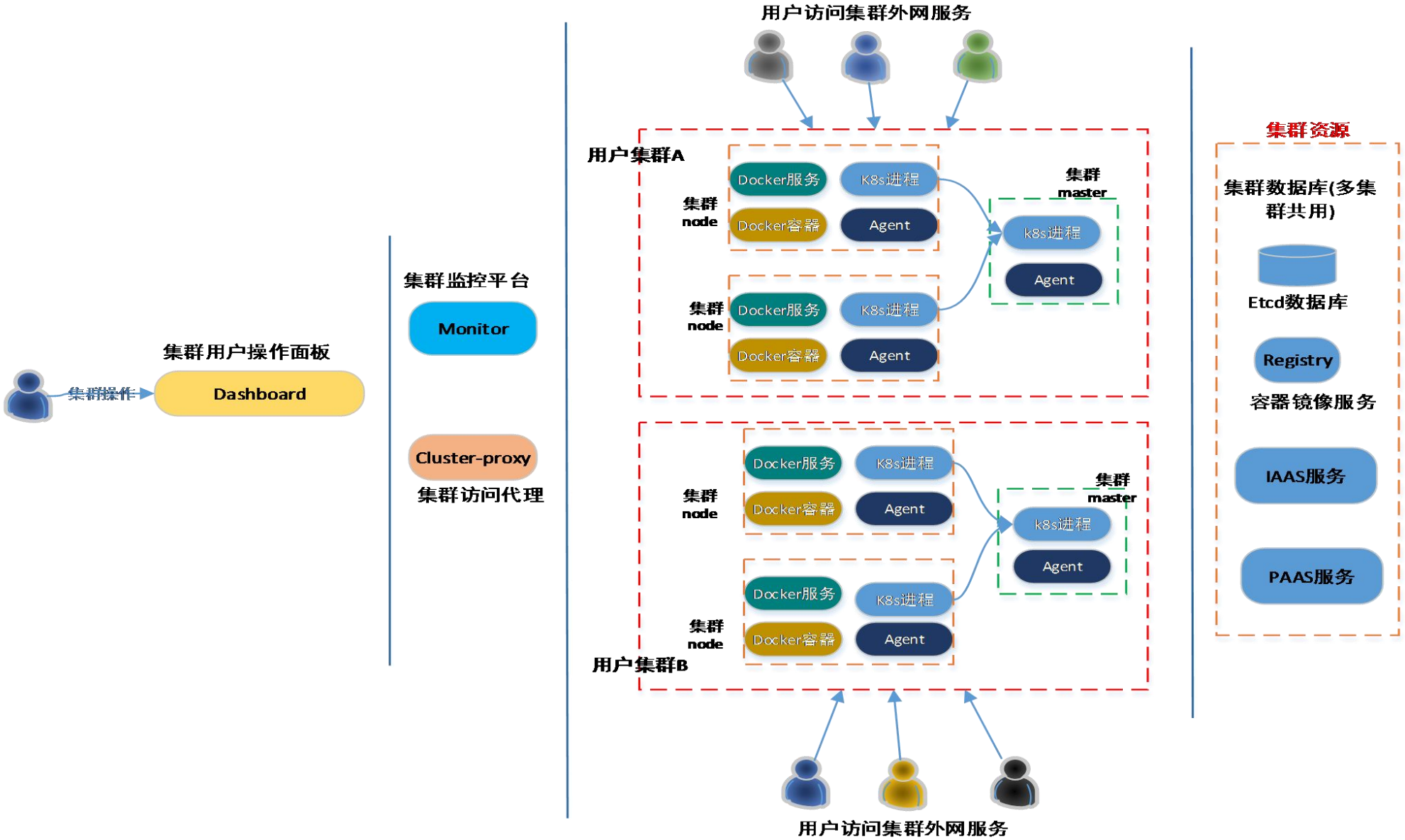
镜像仓库

a、仓库优化，包括添加缓存层以及和cos存储的对接

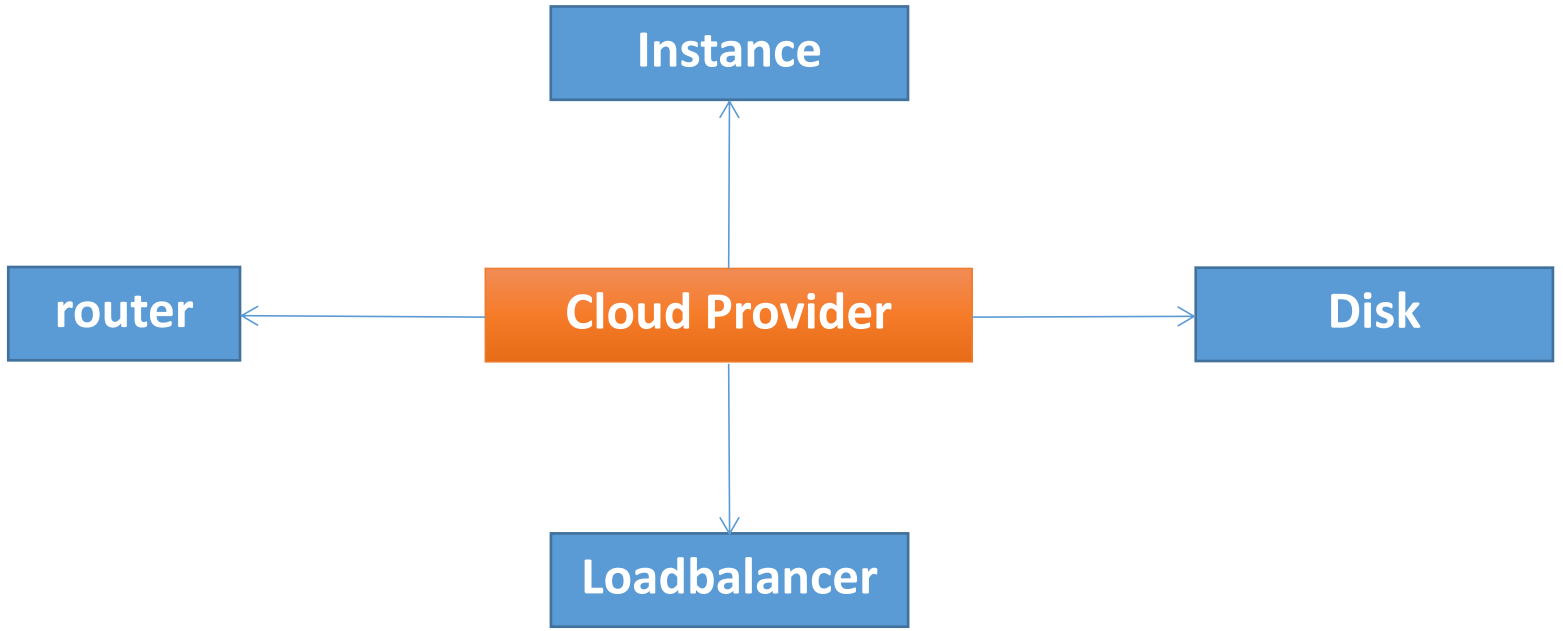
b、CI/CD：代码源对接github，通过镜像触发器可与企业自建CI流程打通

c、提供mirror registry

总体架构


用户访问集群外网服务
集群资源
集群数据库(多集群共用)
Etcd数据库
Registry
容器镜像服务
IAAS服务
PAAS服务
集群监控平台
Monitor
集群用户操作面板
集群操作
Dashboard
Cluster-proxy
集群访问代理
用户集群A
集群node
Docker服务
K8s进程
Docker容器
Agent
集群master
k8s进程
Agent
用户集群B
用户访问集群外网服务

k8s cloud provider

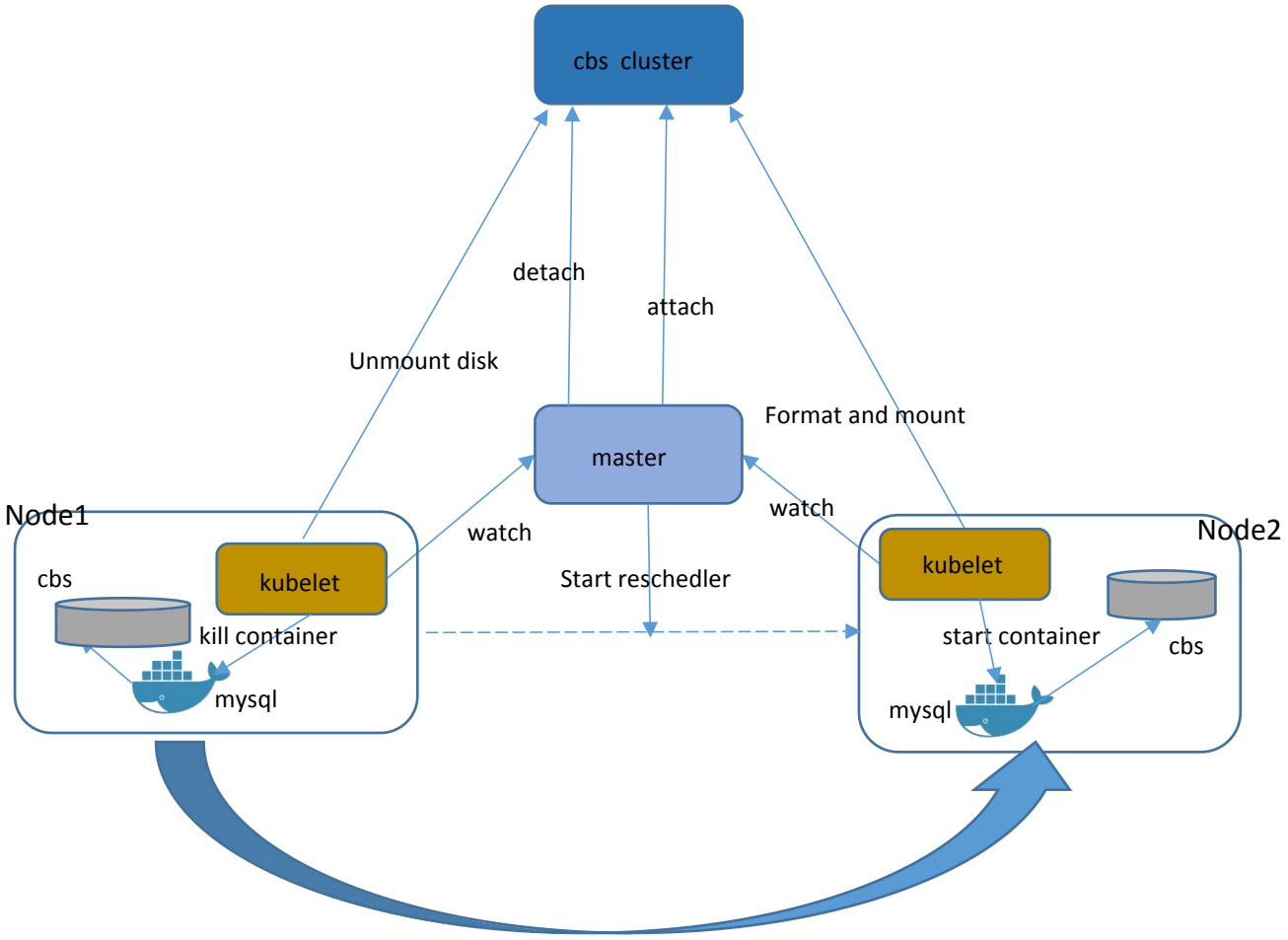


- router:集群新增节点时添加容器子网在哪台VM上的路由信息
- instance:云主机相关信息查询，在kubectl describe node中可看到
- loadbalancer：当service 的spec.type: LoadBalancer时使用
- Disk:cbs盘attach 、detach的接口实现，供volume和storage class 使用
- 使用cloud provider时，apiserver，controller-manager和kubelet的 --cloud-config和--cloud-provider，需配置成对应云厂家

支持cbs的volume

```yaml
spec:
  containers:
  - image: nginx
    imagePullPolicy: Always
    name: nginx
    resources:
      requests:
        cpu: 200m
    securityContext:
      privileged: false
    terminationMessagePath: /dev/termination-log
    terminationMessagePolicy: File
    volumeMounts:
    - mountPath: /mnt
      name: test
  dnsPolicy: ClusterFirst
  imagePullSecrets:
  - name: qcloudregistrykey
  restartPolicy: Always
  schedulerName: default-scheduler
  securityContext: {}
  terminationGracePeriodSeconds: 30
  volumes:
  - name: test
    qcloudCbs:
      cbsDiskId: disk-echj2drf
      fsType: ext4
```

使用cbs的yaml文件



Pod rescheduler

cbs storage class

```yaml
apiVersion: storage.k8s.io/v1beta1
kind: StorageClass
metadata:
  annotations:
    storageclass.beta.kubernetes.io/is-default-class: "true"
  name: cbs
parameters:
  type: cbs
provisioner: cloud.tencent.com/qcloud-cbs
```

使用cbs 的storage class

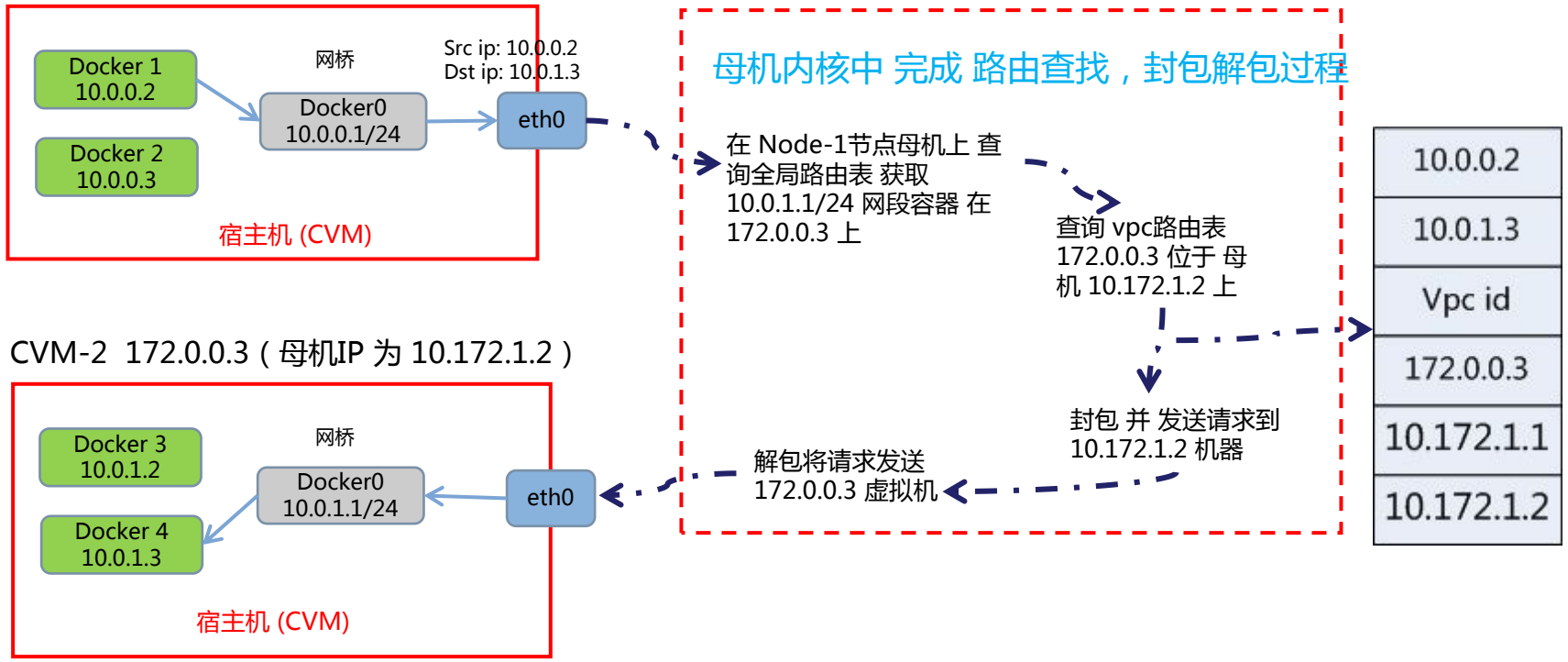
```yaml
  spec:
    terminationGracePeriodSeconds: 10
    serviceAccountName: mongo
    containers:
      - name: mongo
        image: mongo
        command:
          - mongod
          - "--replSet"
          - rs0
          - "--smallfiles"
          - "--noprealloc"
        ports:
          - containerPort: 27017
        volumeMounts:
          - name: mongo-persistent-storage
            mountPath: /data/db
volumeClaimTemplates:
- metadata:
    name: mongo-persistent-storage
    annotations:
      volume.beta.kubernetes.io/storage-class: "cbs"
  spec:
    accessModes: [ "ReadWriteOnce" ]
    resources:
      requests:
        storage: 10Gi
```

mongodb使用cbs storage class的yaml文件

容器之间互访

宿主机 VPC网段：172.0.0.0/24
容器 VPC网段：10.0.0.0/14

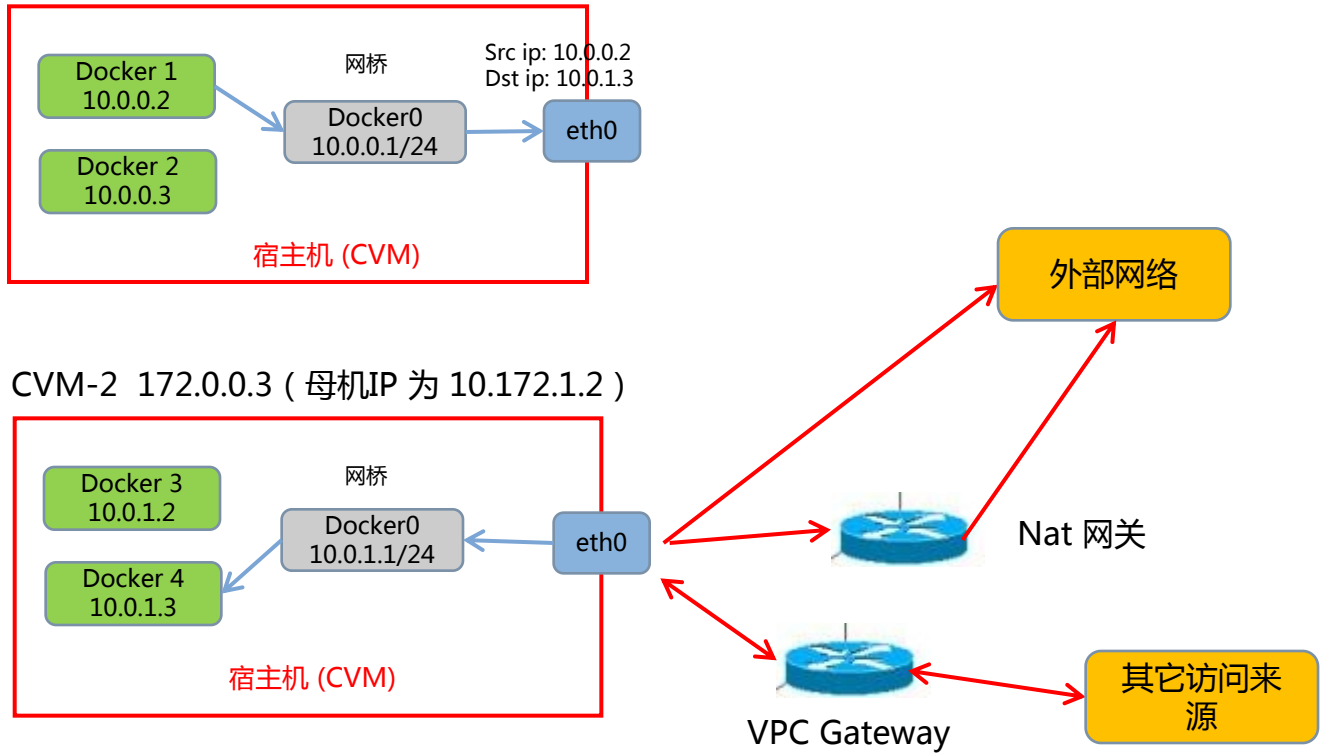
CVM-1 :172.0.0.2 （母机IP 为 10.172.1.1 ）



容器访问其他网络

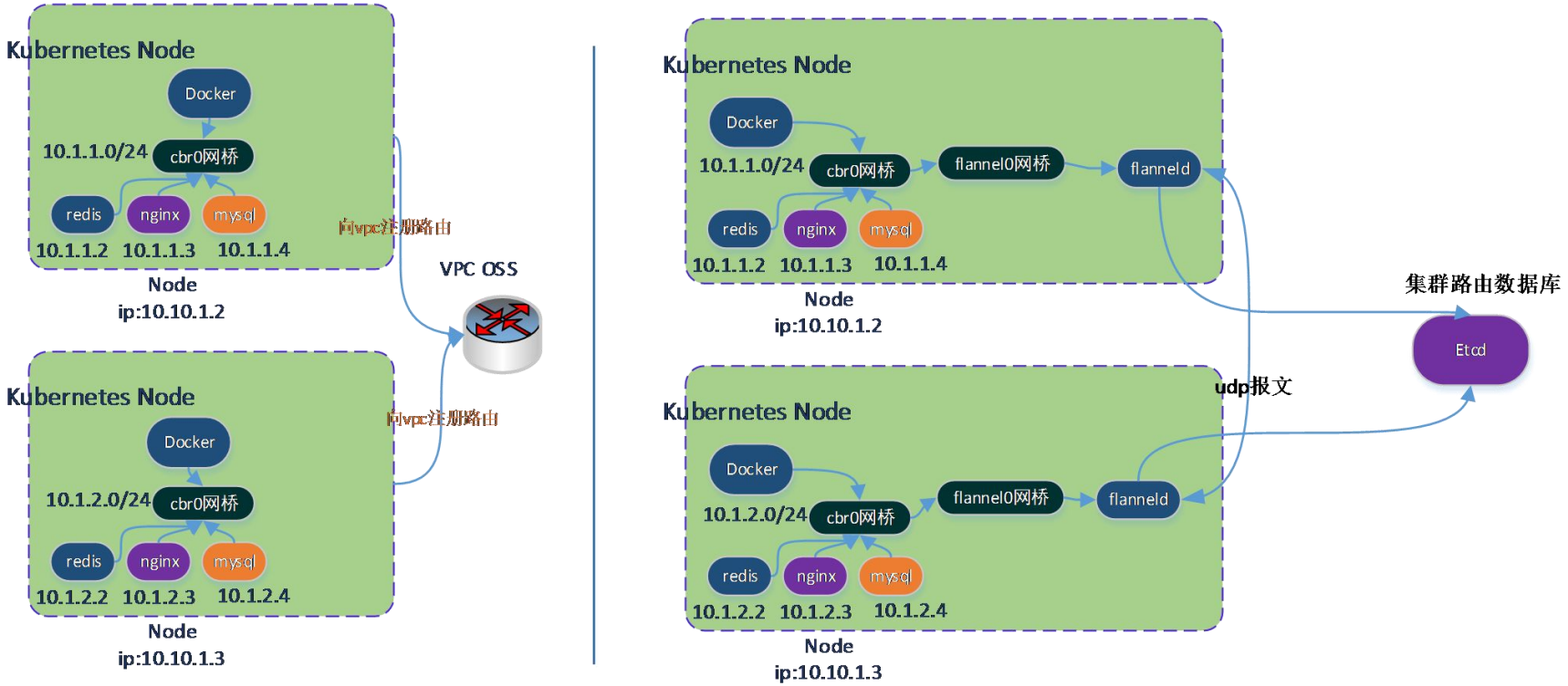
宿主机 VPC网段：172.0.0.0/24
容器 VPC网段：10.0.0.0/14

CVM-1 :172.0.0.2 （母机IP 为 10.172.1.1）



- 容器可以通过 VPC Gateway， 与其它VPC网段，基础网络相互通信
- 容器可以通过宿主机的外网IP，直接访问外网，源地址做NAT转换
- 宿主机没有外网IP的话，容器可以通过NAT网关直接访问外网，源地址做NAT转换

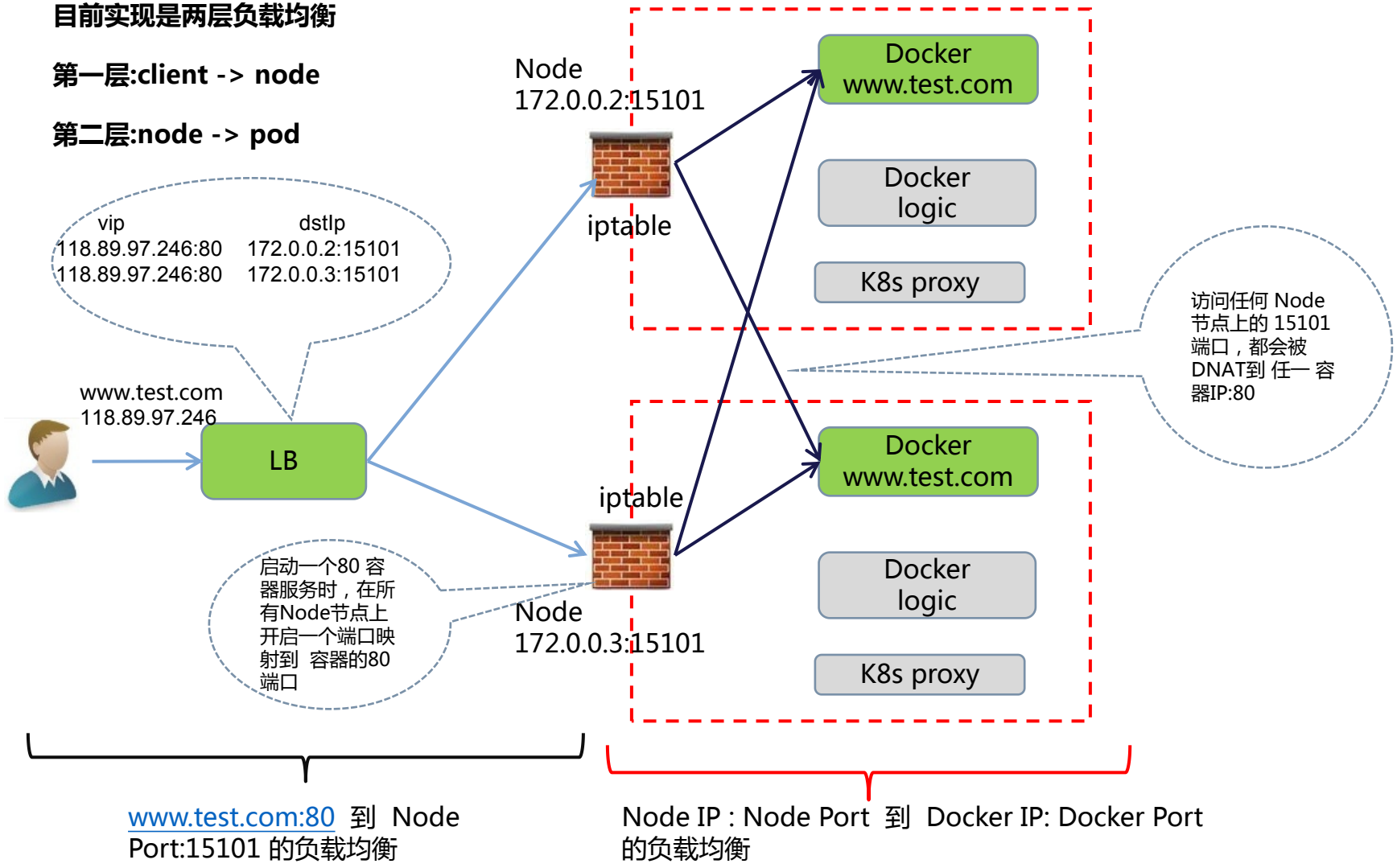
容器网络对比云上自建flannel



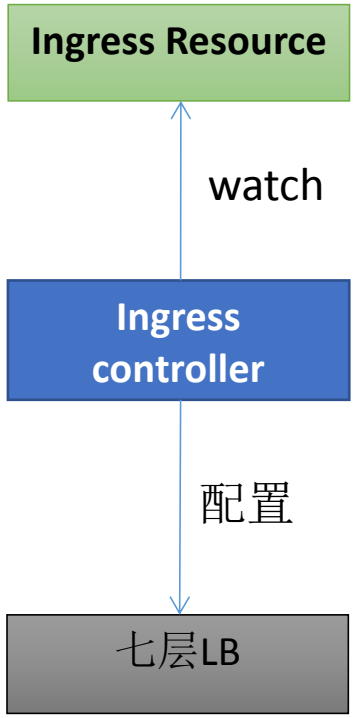
容器路由由vpc软交换机负责

- 所有发往10.1.1.0/24下的流量，全路由到node1
- 无封包解包消耗

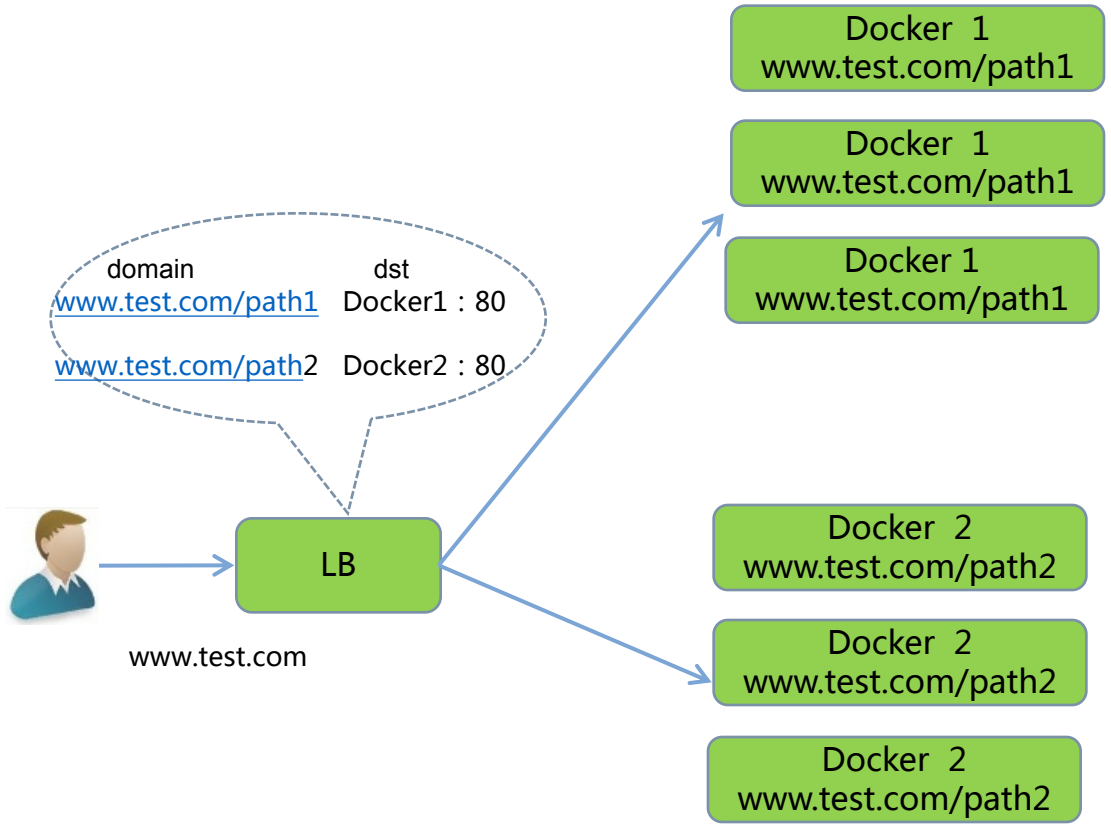
对接四层LB


LB 插件 将K8S 与 腾讯云负载均衡打通,
目前实现是两层负载均衡
第一层:client -> node
第二层:node -> pod
Node
172.0.0.2:15101
Docker
www.test.com
Docker
logic
K8s proxy
iptable
vip
dstIp
118.89.97.246:80
172.0.0.2:15101
118.89.97.246:80
172.0.0.3:15101
www.test.com
118.89.97.246
LB
访问任何 Node 节点上的 15101 端口，都会被 DNAT到 任一 容器IP:80
iptable
Docker
www.test.com
Docker
logic
K8s proxy
启动一个80 容器服务时，在所有Node节点上开启一个端口映射到 容器的80端口
Node
172.0.0.3:15101
www.test.com:80 到 Node Port:15101 的负载均衡
Node IP : Node Port 到 Docker IP: Docker Port 的负载均衡

对接七层LB

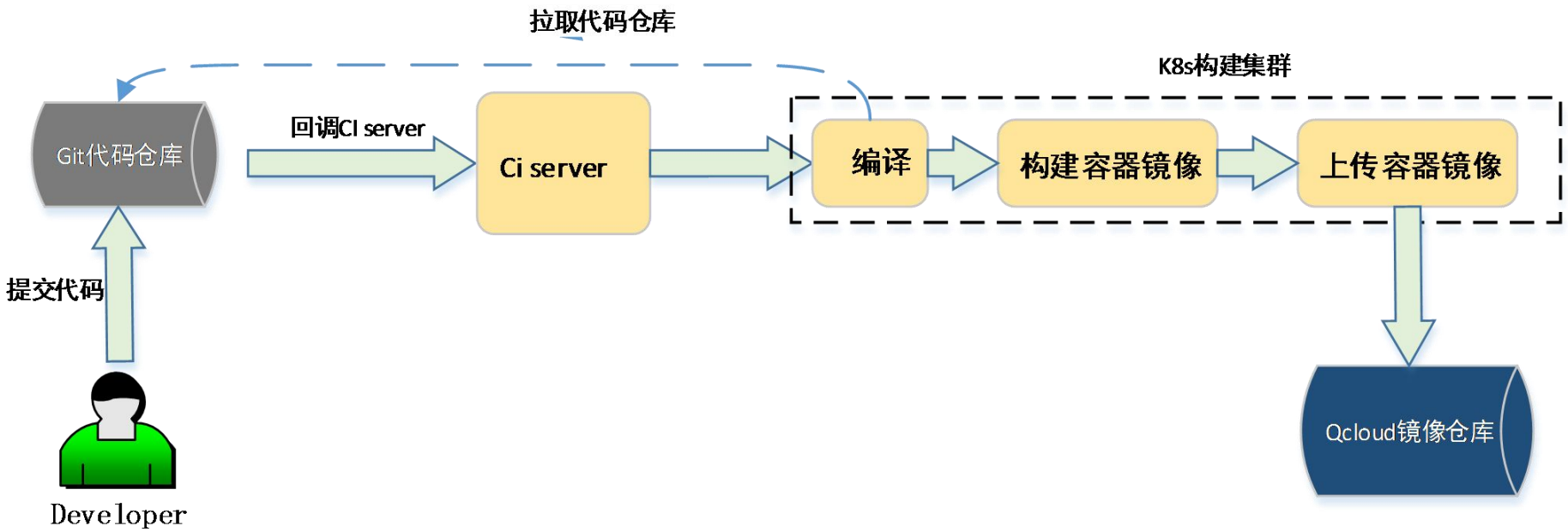


ingress 原理



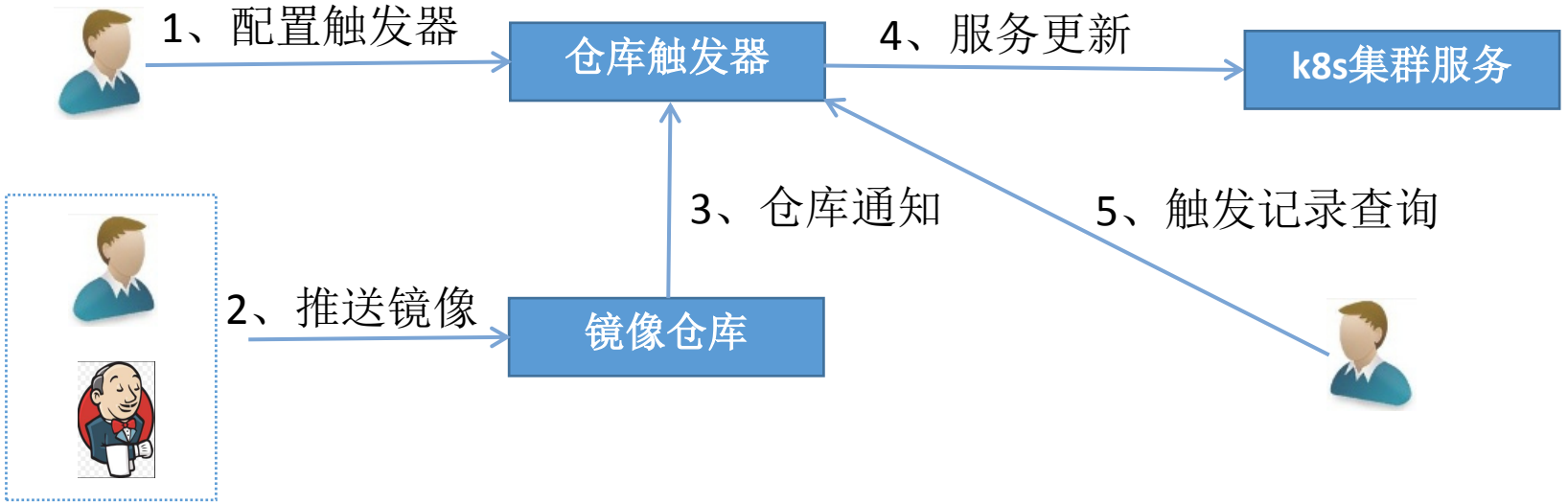
根据域名路径转发到 后端不同 容器服务

CI实现



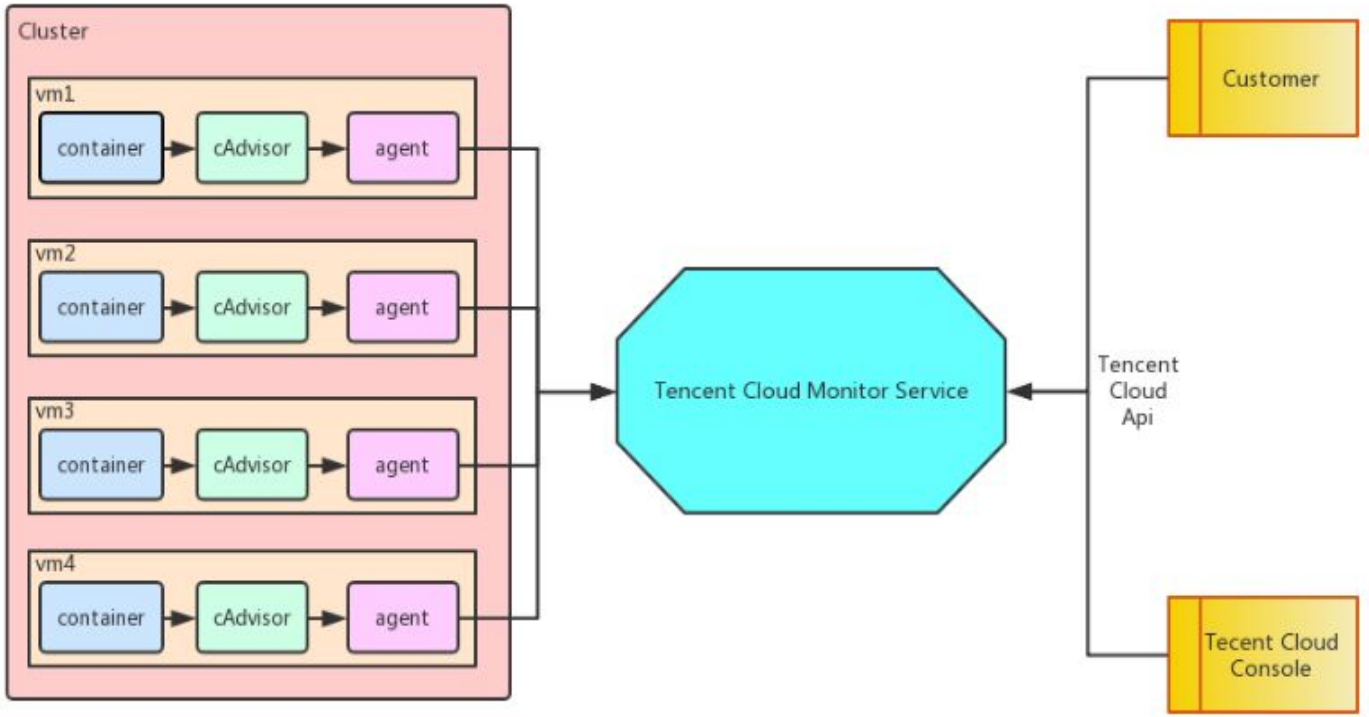
- 支持Github和gitlab仓库，在仓库提交代码后，在我们的CI平台上自动触发容器镜像构建。
- 支持在控制台上传dockerfile来构建容器镜像。
- 构建任务在海外执行，访问海外资源(基础镜像、安装包等)速度很快。
- 支持并发执行，用户默认2个并发构建任务，每个构建任务默认分配2C2G资源，构建时间限制为40Min。

CD实现



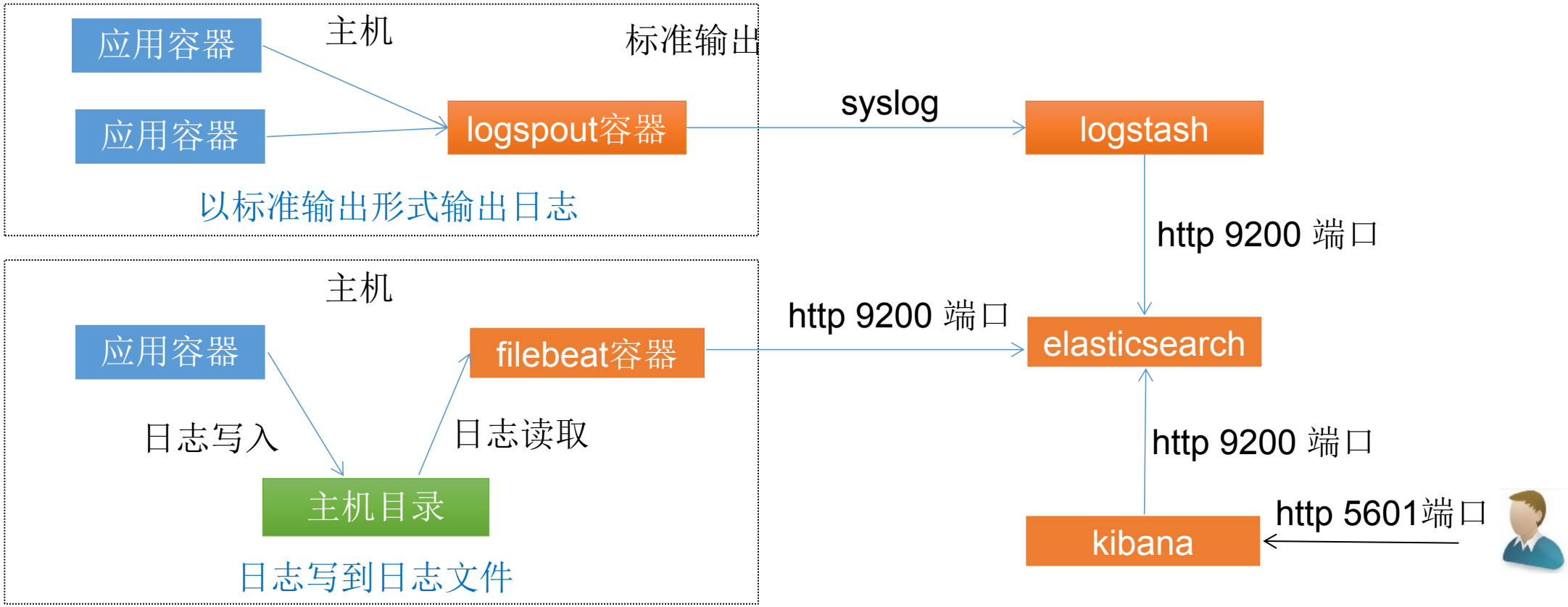
- 触发器配置包括触发条件，触发动作设置
- 支持手动和CI工具触发
- 支持CD结果查询，可查询触发动作是否成功及相关触发参数

监控实现



- agent每分钟会从cadvisor采集一次数据，报给监控平台
- 支持的指标，包括主机、容器和服务层面的监控数据，如cpu、内存、磁盘和网络

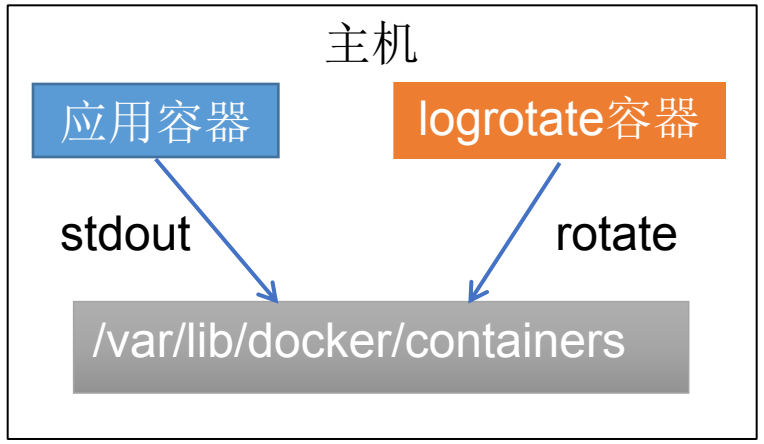
日志收集


应用容器
主机
标准输出
应用容器
logspout容器
syslog
logstash
以标准输出形式输出日志
http 9200 端口
主机
http 9200 端口
应用容器
filebeat容器
elasticsearch
日志写入
日志读取
http 9200 端口
主机目录
http 5601端口
kibana
日志写到日志文件

本地日志清理

docker日志默认驱动

清理方式1:daemon set方式部署logrotate容器



清理方式2:修改dockerd 启动参数

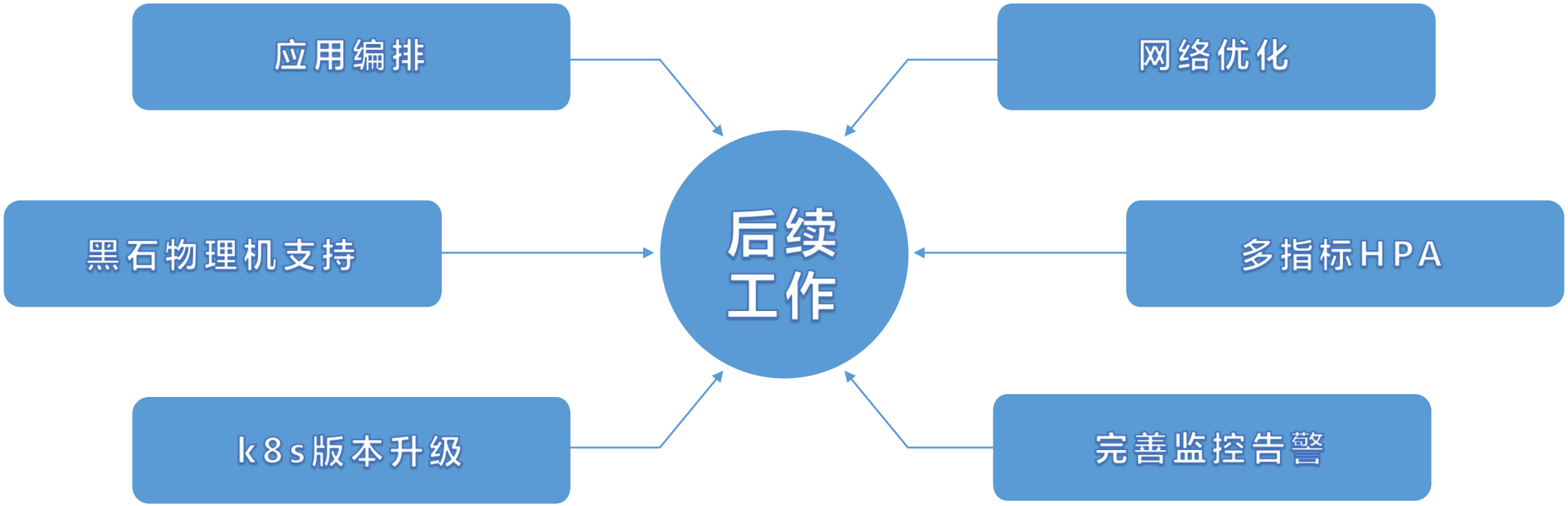
1、创建/etc/dockerd/daemon.json

```
{
  "log-driver":"json-file",
  "log-opts":{
            "max-size" :"10m","max-file":"3"
    }
}
```

2、修改dockerd 服务配置文件
/etc/systemd/system/multi-user.target.wants/dockerd.serviced
添加dockerd启动参数--config-file=/etc/docker/daemon.json

3、重启dockerd服务

后续工作



Thanks