Kubernetes

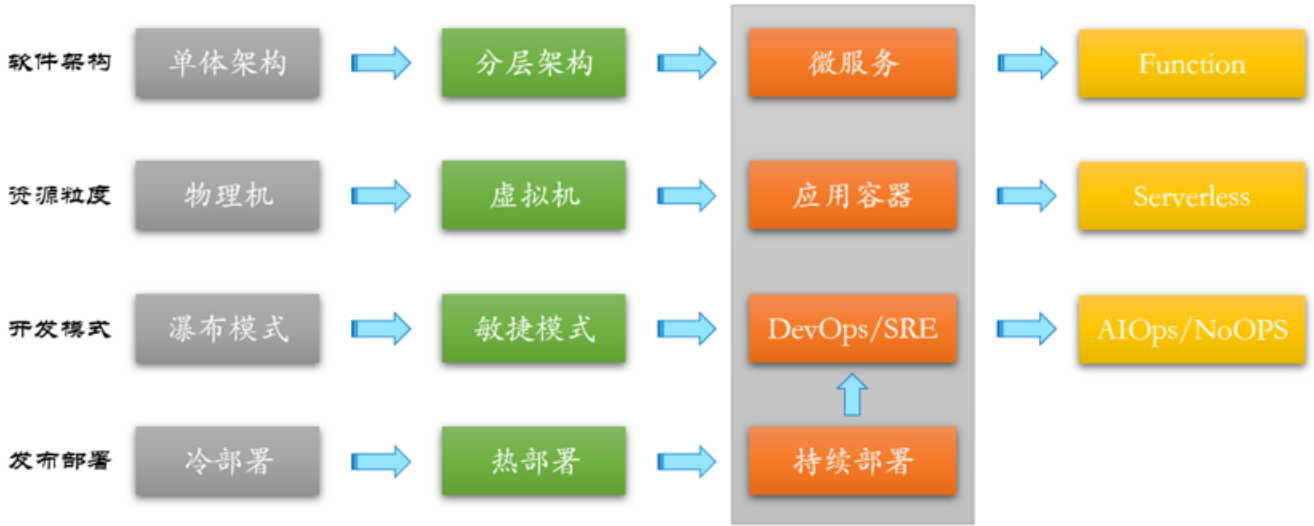
主讲：马永亮

http://www.magedu.com
http://github.com/ikubernetes

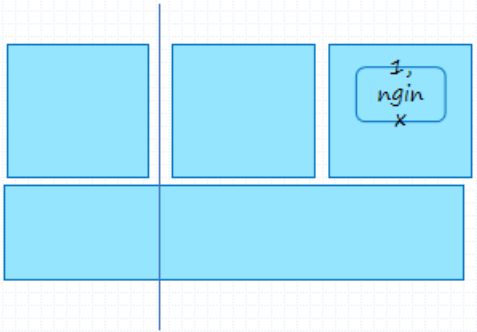
个人介绍

- 马哥教育创始人&CEO
- 著有《Kubernetes进阶实战》一书，正在改版第二版，几乎完全重写，并添加服务Envoy和服务网格的话题；
- Linux系统运维、云计算、大数据和运维开发领域践行者和布道师
- 热爱开源技术，维护的Github页面 https://github.com/ikubernetes

IT技术发展趋势



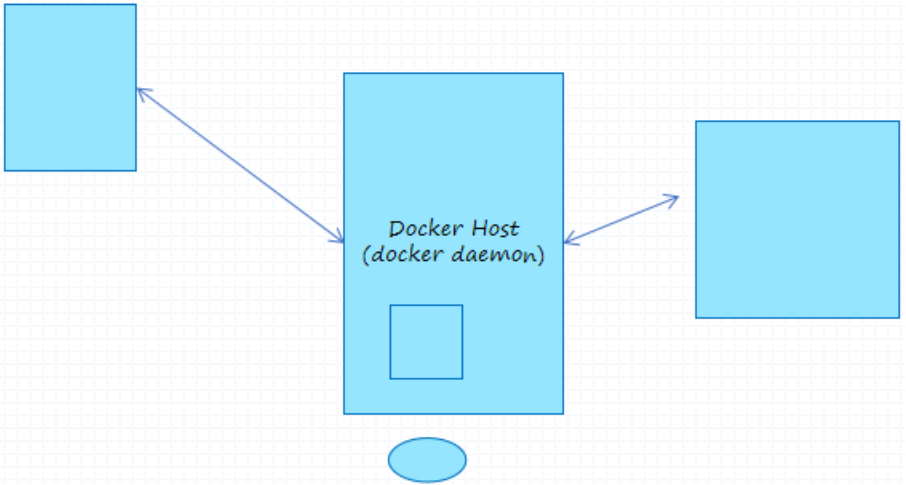
- 底层平台，Hardware, OS,
- 镜像: 分层构建，aufs, overlayfs2,



repo:tag, Docker Hub, Quey

- container, image, network, volume, ...
- docker-compose

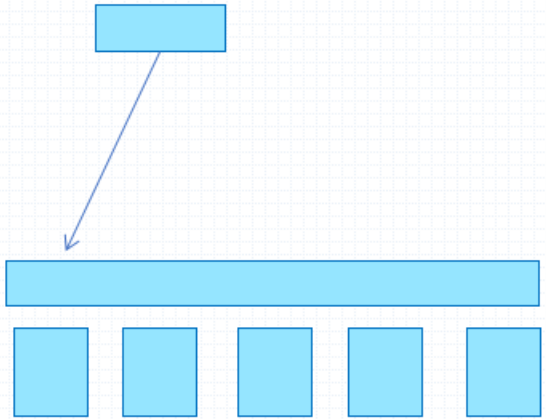
- Network Model
- none, bridge, host,
- overlay (vxlan, vlan)



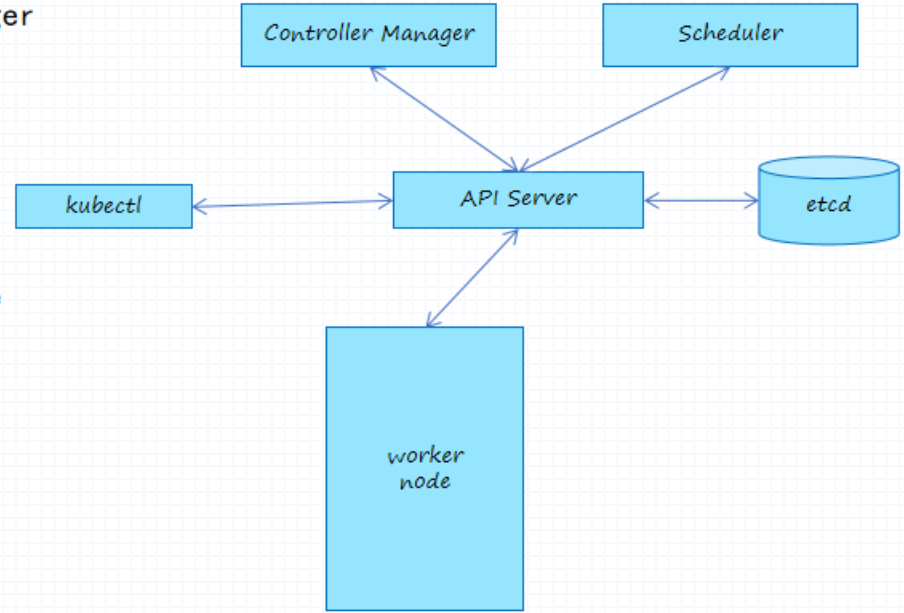
- k8s, swarm, mesos (DC OS), marathon
- Borg, kubernetes, k8s, SDN, 2015,
- CNCF, Istio

- Landscape

operator, controller

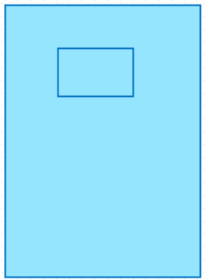


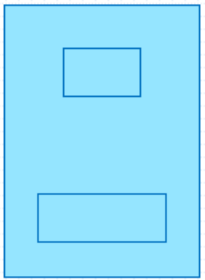
- master node
  - API Server
  - Controller Manager
  - Scheduler
  - Etcd
- worker node
  - kubelet
  - kube-proxy
  - container engine
- worker node

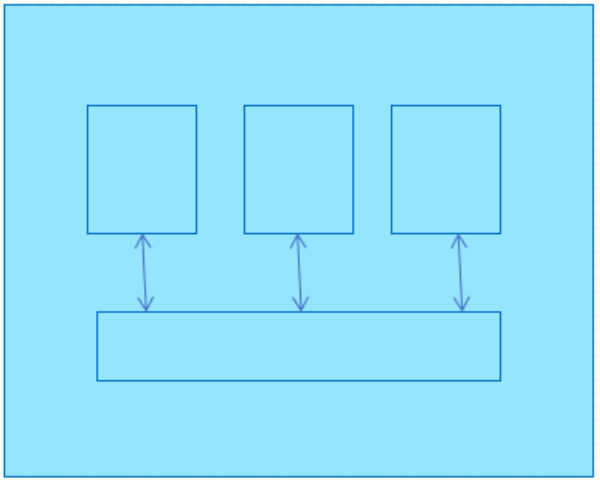


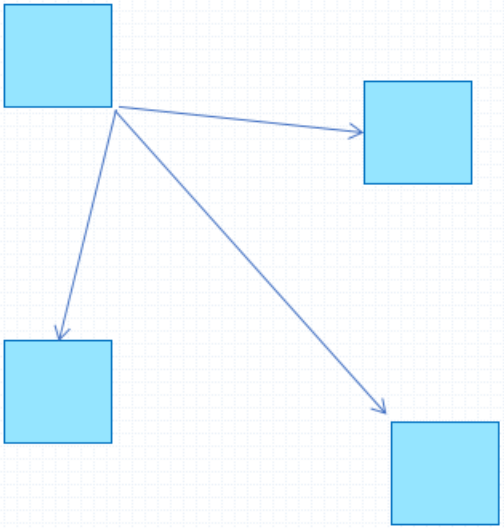
172.17.0.1/16

- bridge



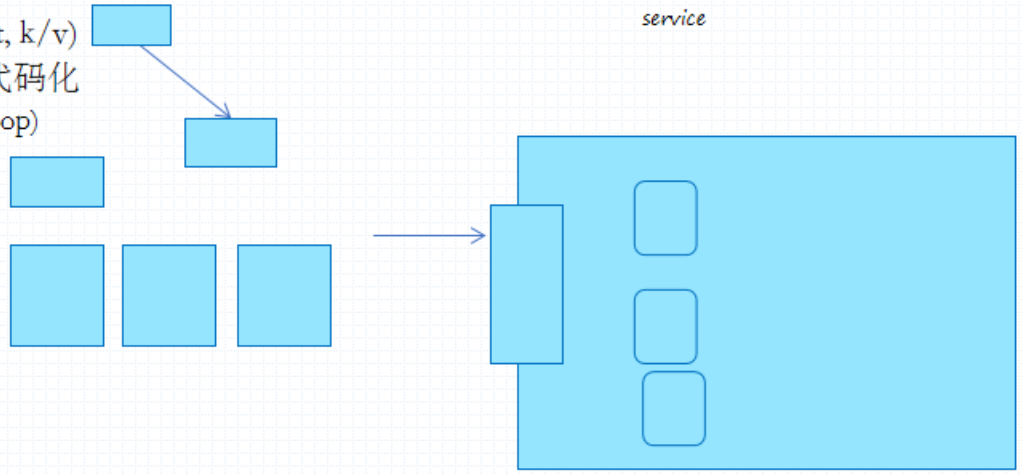






https, 6443， PaaS

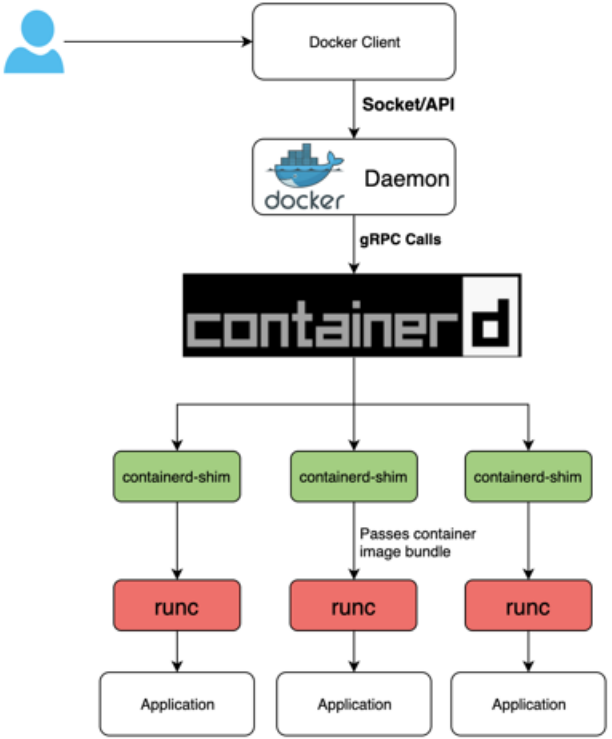
- MiniKube, 分布式Kube Cluster
  - kubeadm, Master, Worker
  - API Server -> etcd(raft, k/v)
  - Controller Manager, 代码化
    - Controller (control loop)
  - Scheduler, Pod
- Node
  - kubelet, agent
  - Docker
  - kube-proxy
- Add-ons
  - KubeDNS: CoreDNS
  - Dashboard, Web UI
  - 监控系统：prometheus
  - 集群日志系统：EFK, LG
  - Ingress Controller：



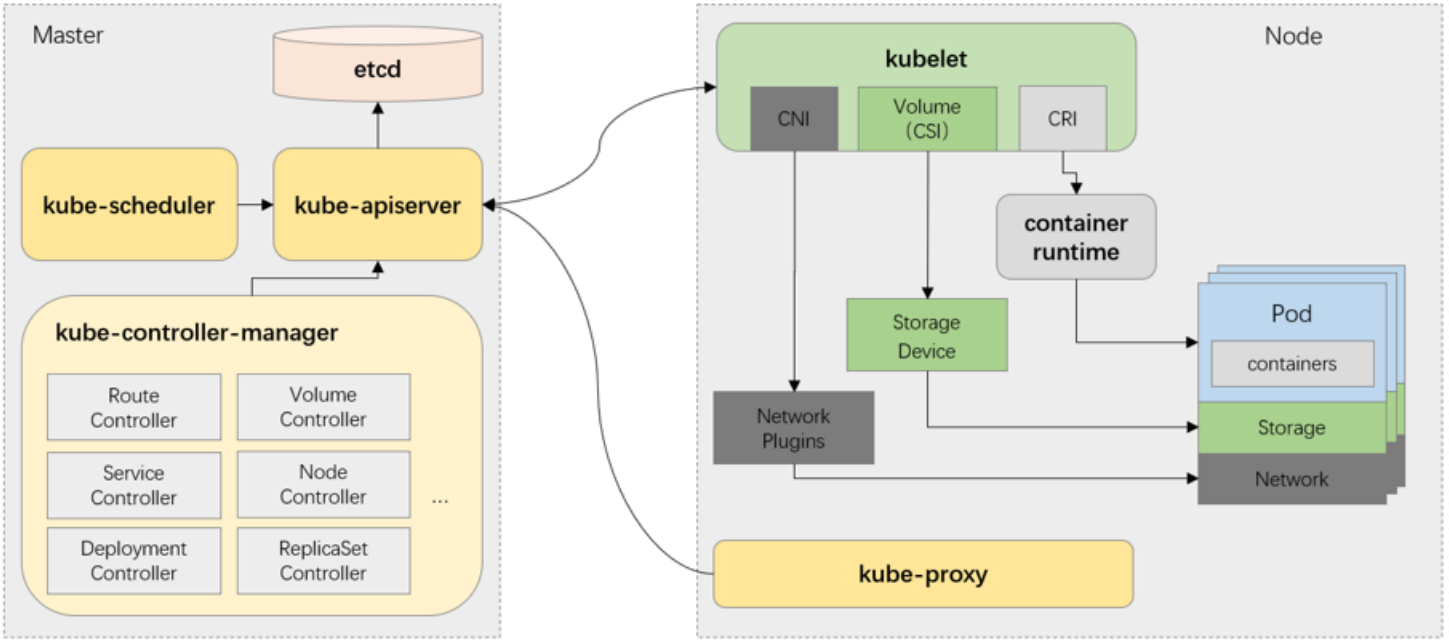
资源规范，配置清单

◆YAML，JSON

```
apiVersion: ...    # 资源对象所属的API群组及版本
kind: ...          # 资源类型
metadata:          # 资源对象的原数据
  ...
spec:              # 所需状态，或称为期望状态
  ...
```



gcr.io

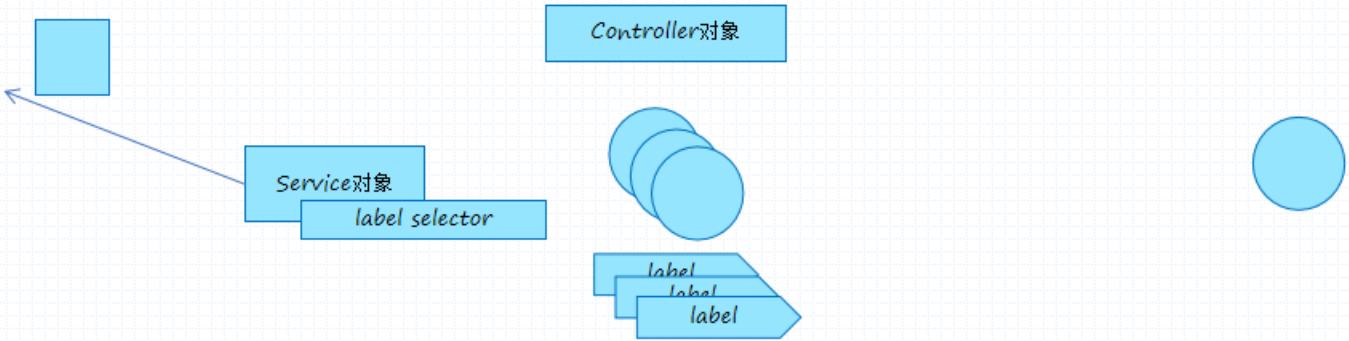


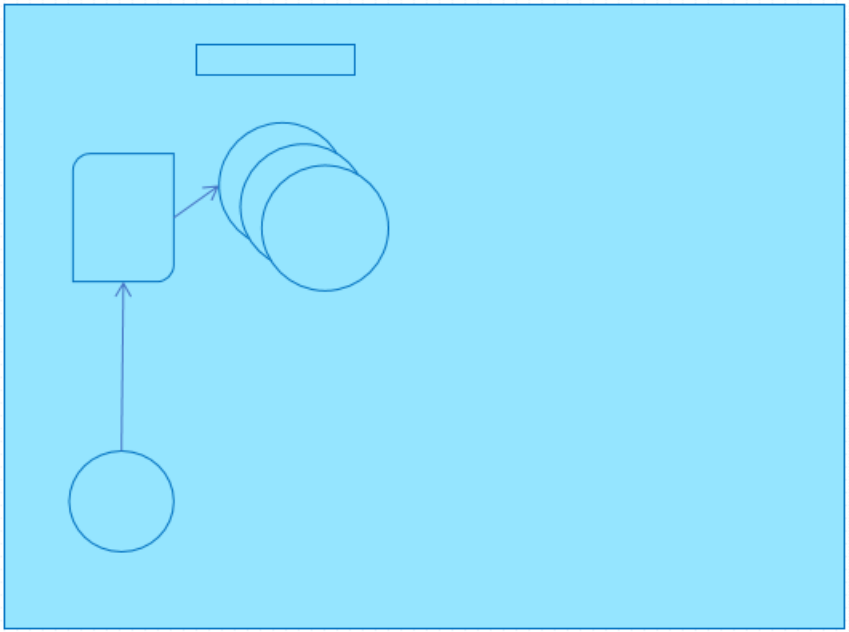
- CNI：Container Network Interface
  - flannel
  - Project Calico
  - Canal
- CRI: Runtime, 运行时
- CSI：Storage, 存储

声明式配置

◆运行应用：

- Controller：Nginx，3
- Service

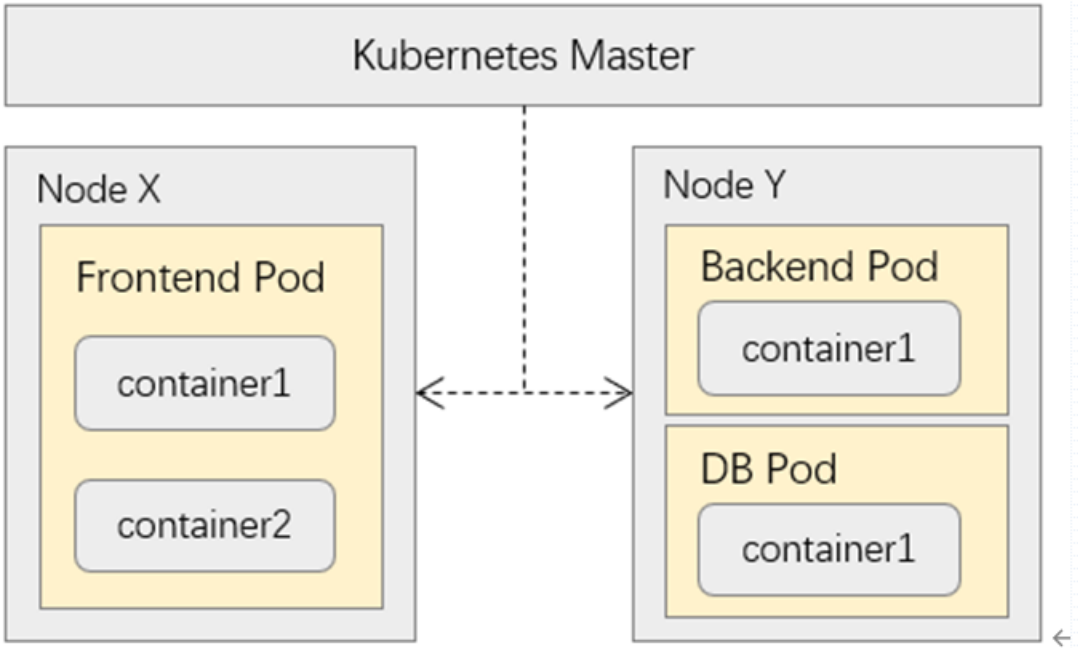


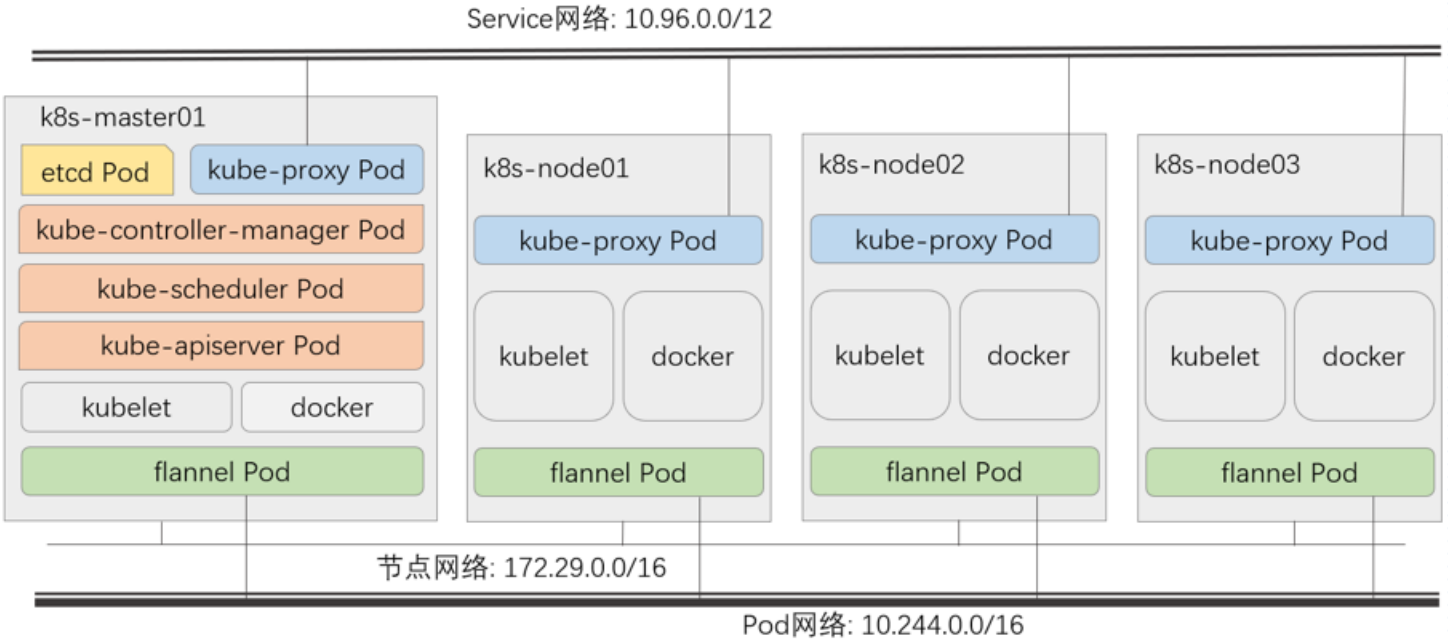


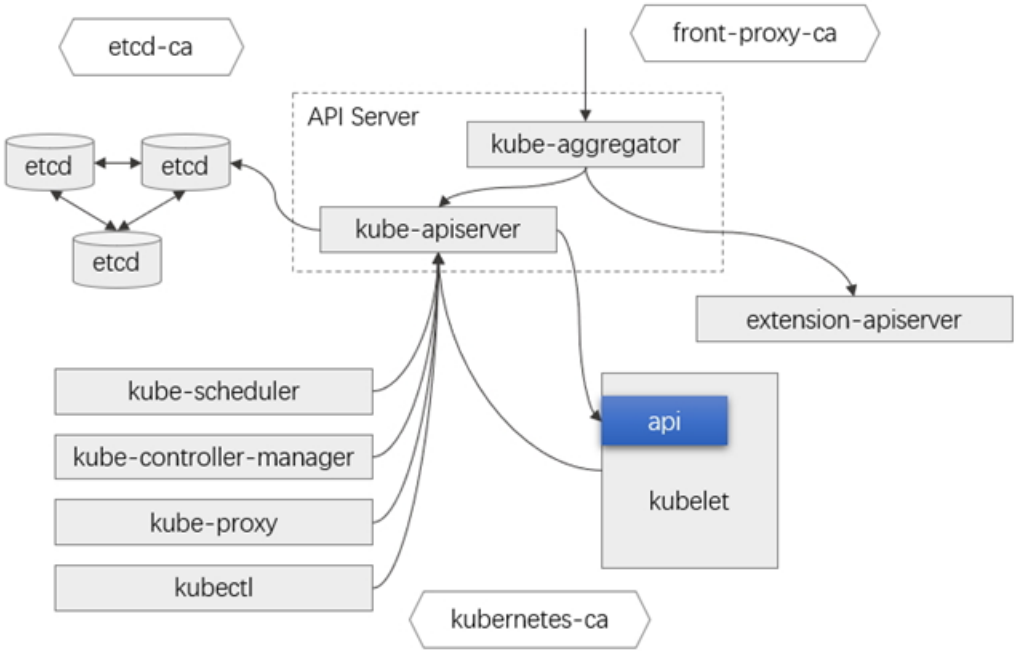
- kubeadm

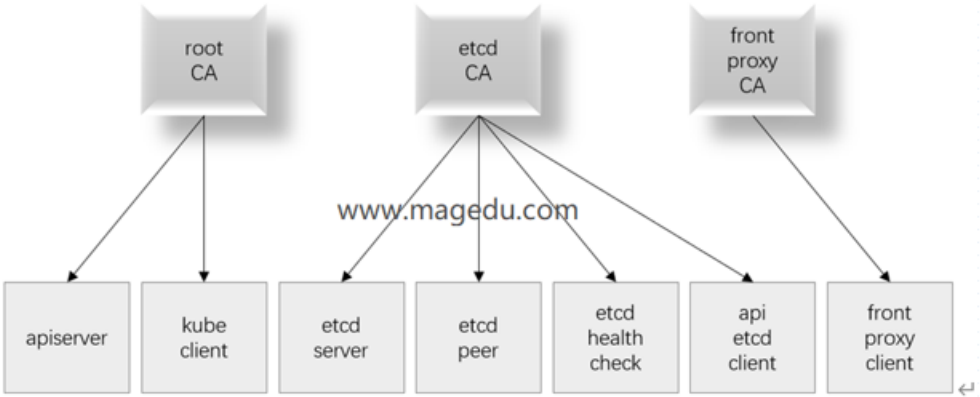
- master
  - kube-apiserver, kube-controller-manager, kube-scheduler, etcd
- worker
  - kubelet, kube-proxy, docker

  - 要点：每个节点都要部署docker, kubelet, kubeadm, kubectl

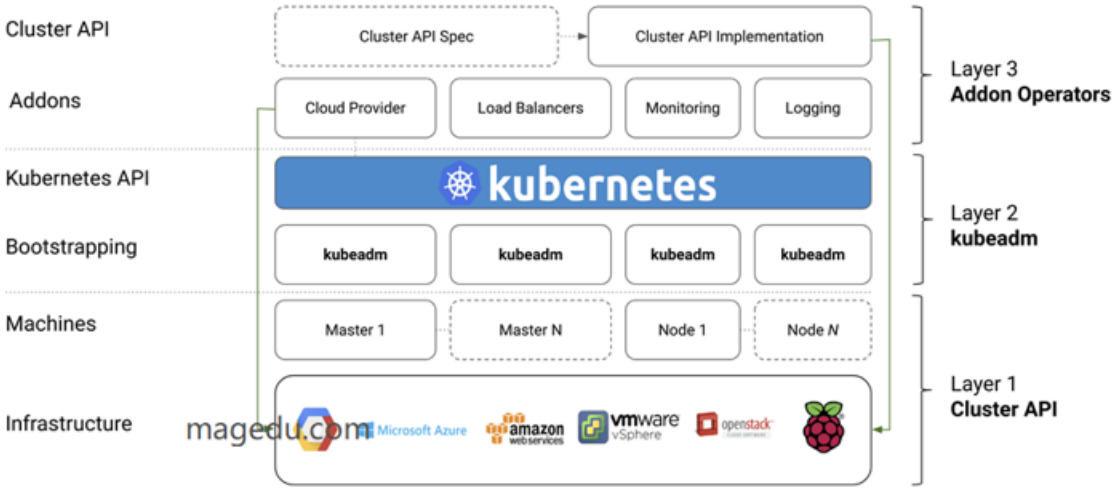




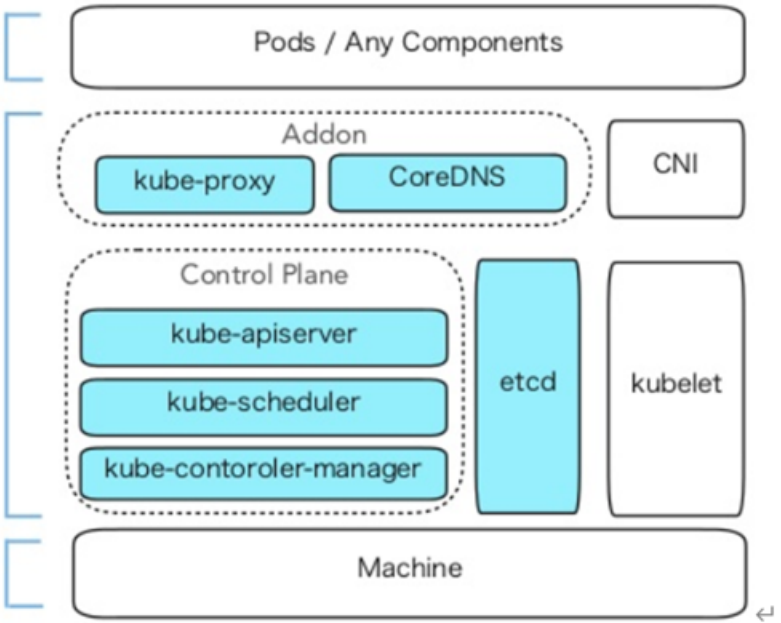


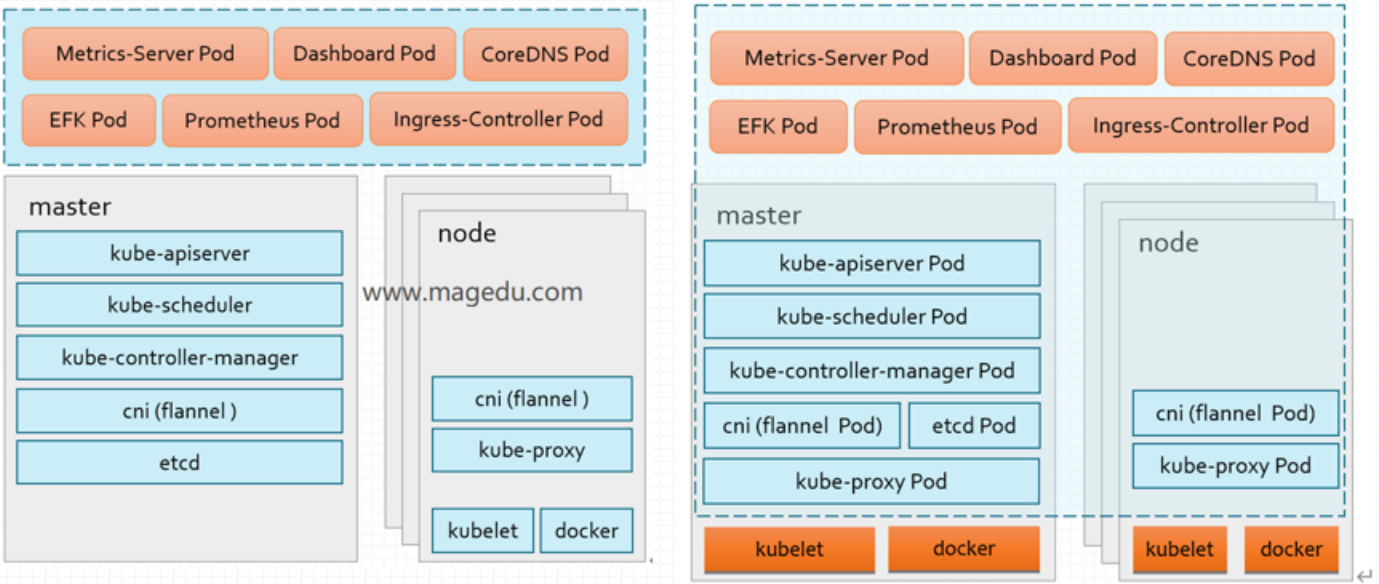

root CA
etcd CA
front proxy CA
www.magedu.com
apiserver
kube client
etcd server
etcd peer
etcd health check
api etcd client
front proxy client

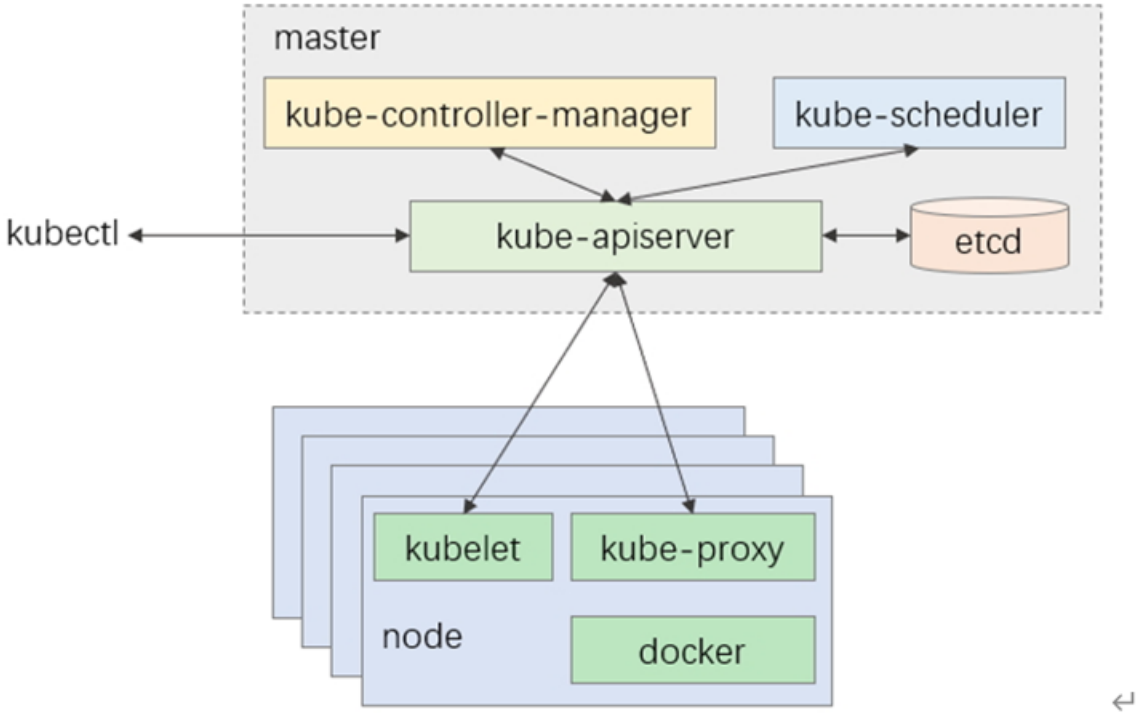
- Kubernetes集群的部署和运行模型
  - 二进制程序
  - Pod：
    - Static Pod
    - Pod

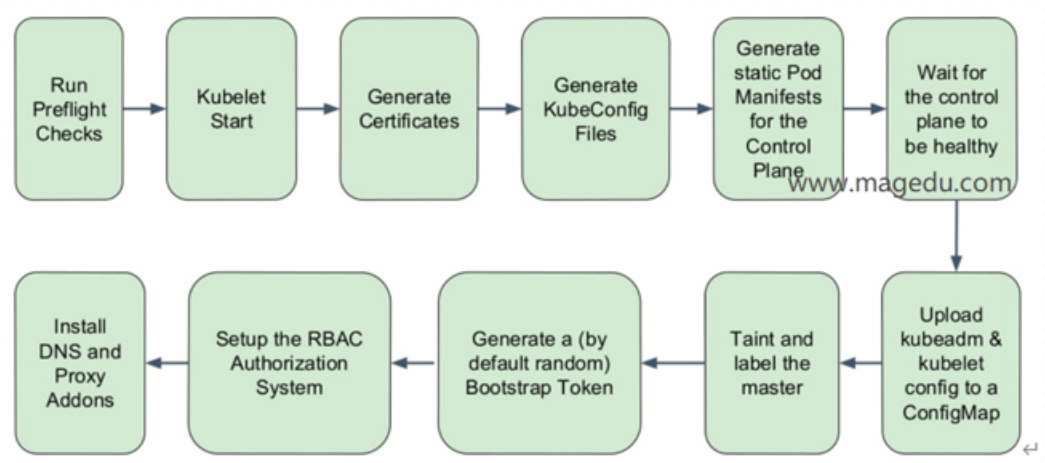


overlay，叠加， underlay，承载



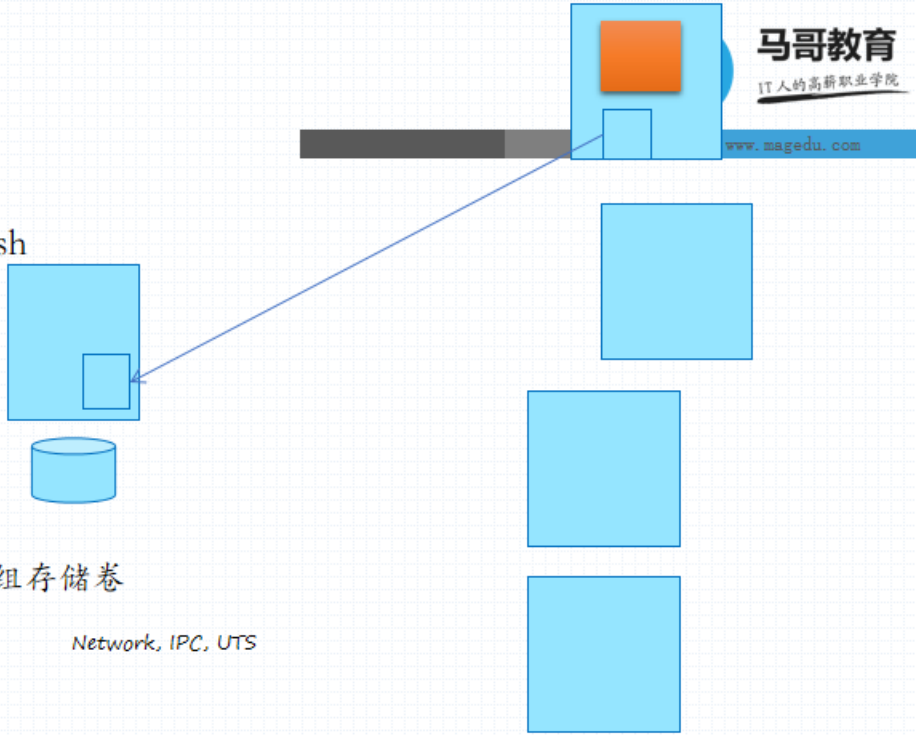





Run Preflight Checks
Kubelet Start
Generate Certificates
Generate KubeConfig Files
Generate static Pod Manifests for the Control Plane
Wait for the control plane to be healthy
www.magedu.com
Upload kubeadm & kubelet config to a ConfigMap
Taint and label the master
Generate a (by default random) Bootstrap Token
Setup the RBAC Authorization System
Install DNS and Proxy Addons

JRuby

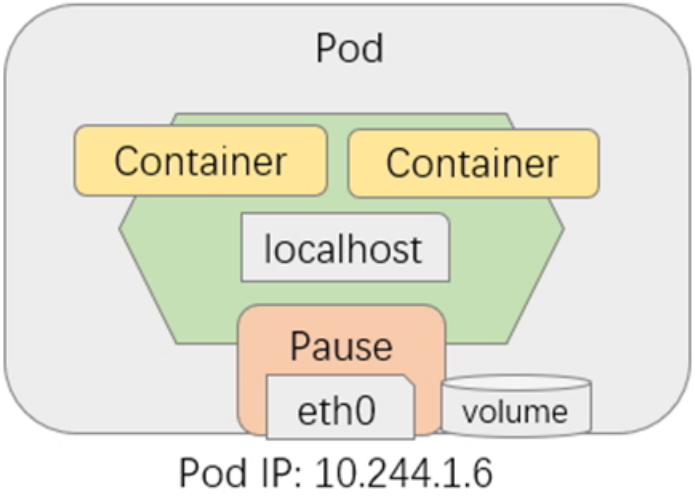
- ElasticSearch
- Filebeat/Fluentd/fluent-bit/logstash
- Kibana
- Loki, Grafana



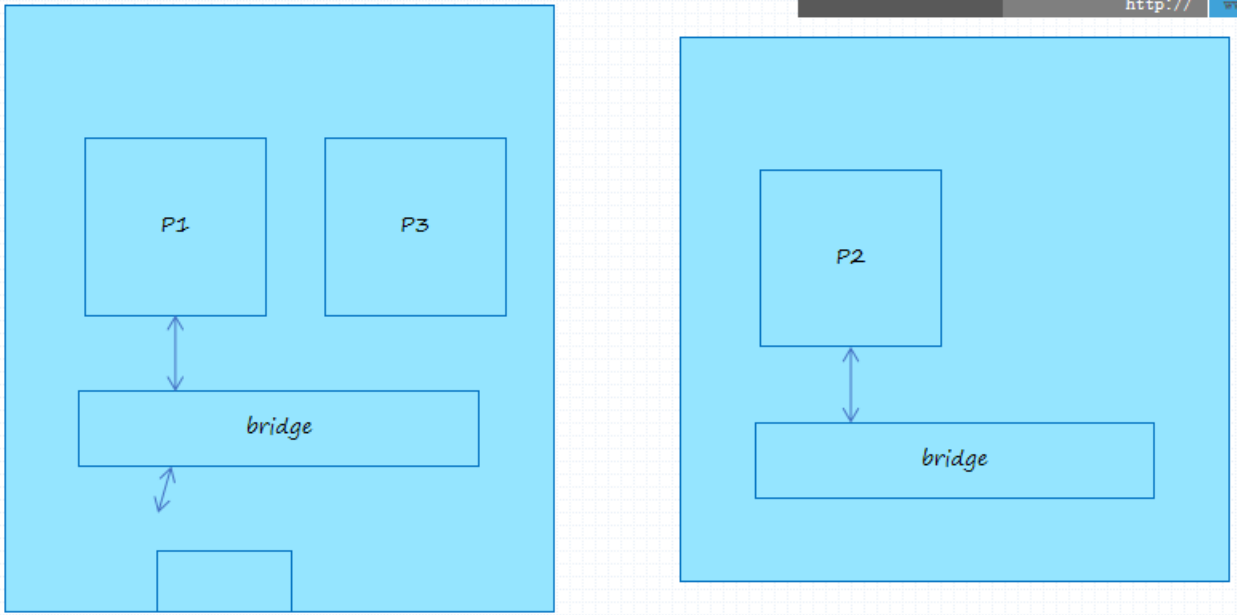
- 容器存在“关系
  - 亲密：Pod，同进同退，共享一组存储卷
    - Container to Container
  - 非紧密：
    - Pod to Pod：网络插件
    - Pod to Service: iptables或ipvs规则
      - ✓ kube-proxy：把集群上的每一个service的定义转换为本地的ipvs或iptables规则
    - External Client -> Service or Pod

Network, IPC, UTS

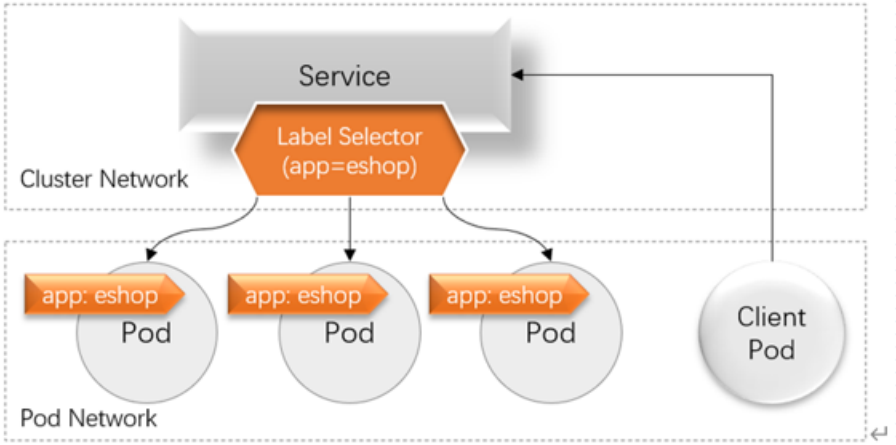
Infrastructure Pod， Pause，暂停



10.108.231.100

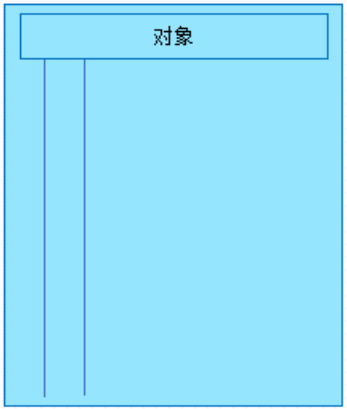


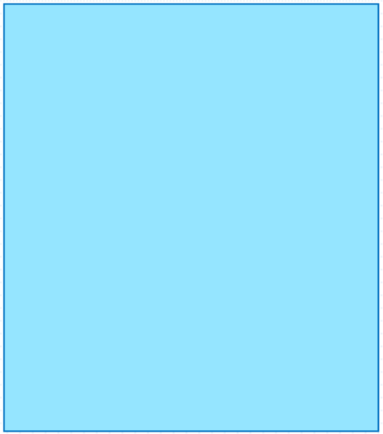
Service



kubectl, KUBE API Server

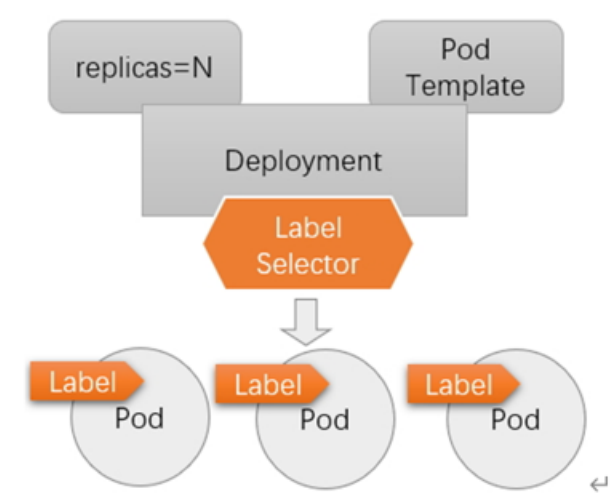
- API Server
  - RESTful，资源resource，
  - 资源类型：Pod，Deployment，Service，PodList，DeploymentList，...

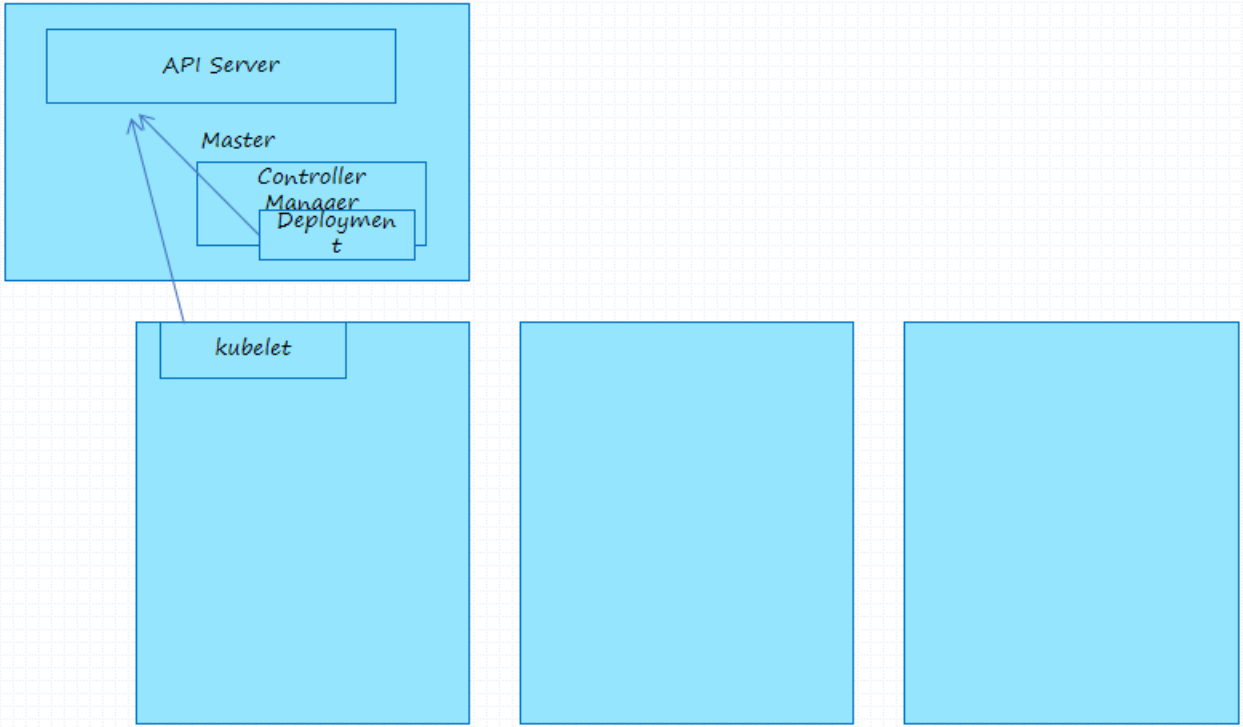


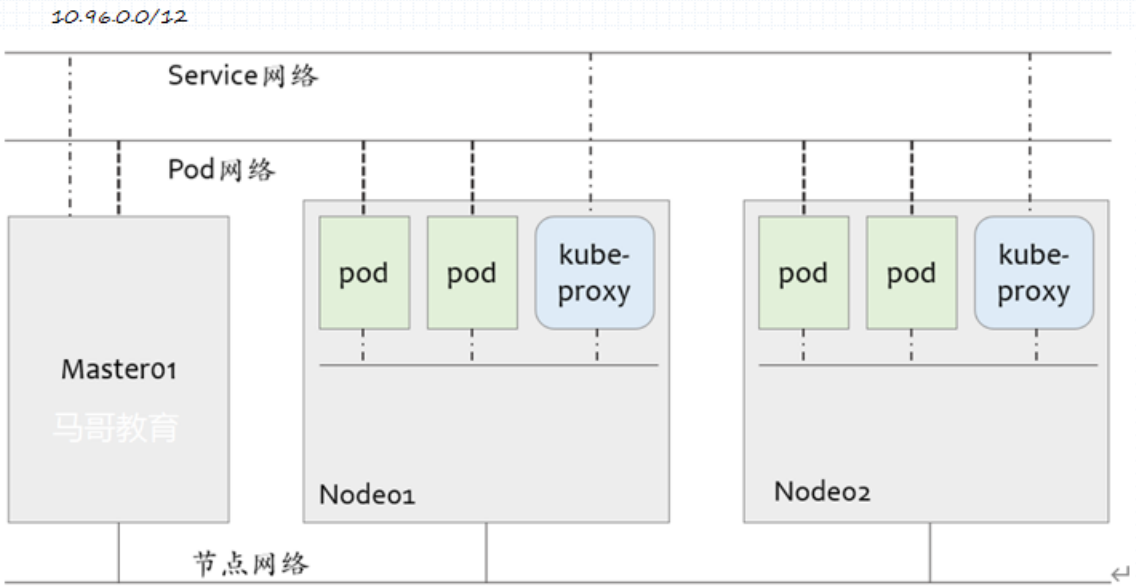


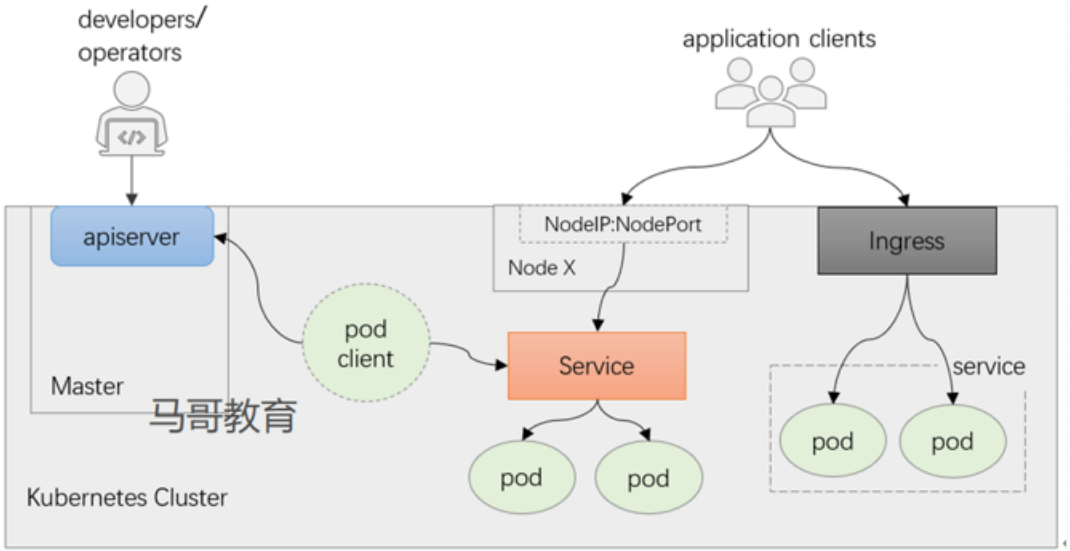
- 命令式命令
  - 命令，及其选项（从选项中读取配置数据）来实现；
- 命令式配置文件
  - 命令，从配置文件加载配置数据
- 声明式配置文件
  - 声明式命令，从配置文件加载配置数据

Pod控制器

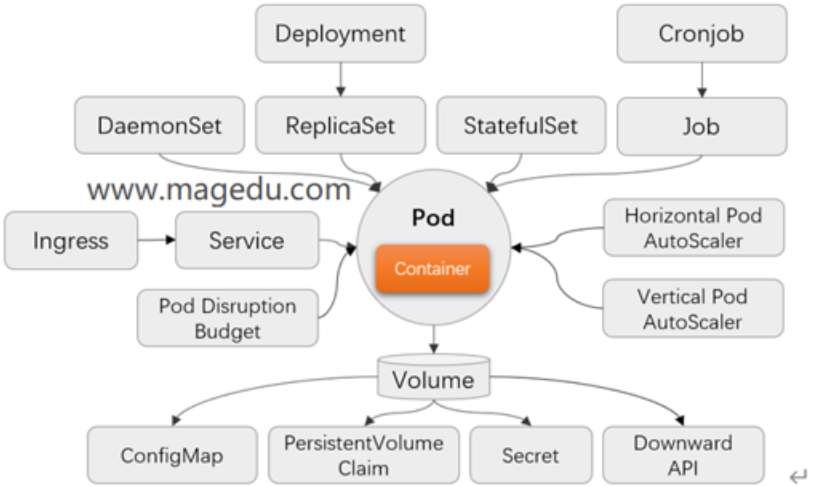

replicas=N
Pod Template
Deployment
Label Selector
Label
Pod
Label
Pod
Label
Pod




10.96.0.0/12
Service网络
Pod网络
Master01
马哥教育
pod
pod
kube-proxy
Node01
pod
pod
kube-proxy
Node02
节点网络

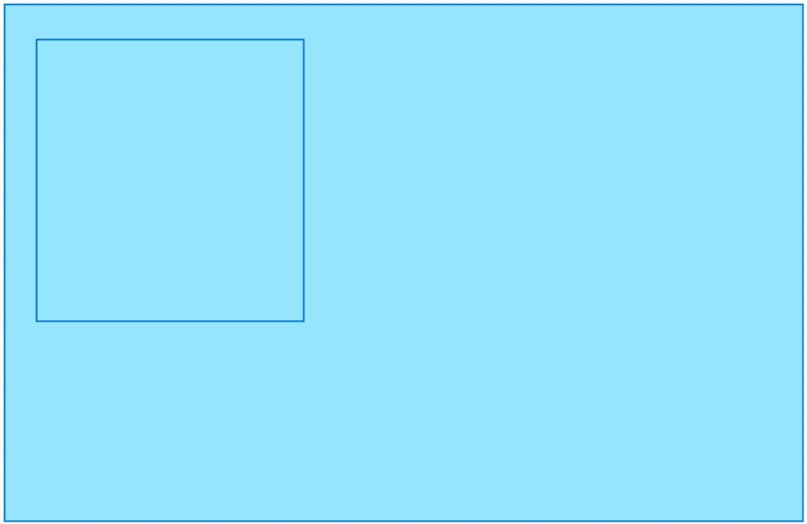

developers/
operators
application clients
apiserver
NodeIP:NodePort
Node X
Ingress
pod
client
Service
service
Master
马哥教育
pod
pod
pod
pod
Kubernetes Cluster

- RESTful API, https
  - 资源类型，schema
    - Pod, Deployment, Service
  - 对象，object
  - 资源规范
    - apiVersion, kind, metadata, spec(期望状态), status(实际状态)
    - 控制器：负责确保将实际状态设定不断逼近或等同于期望状态
    - Control Loop
      - ✓ create, delete, change

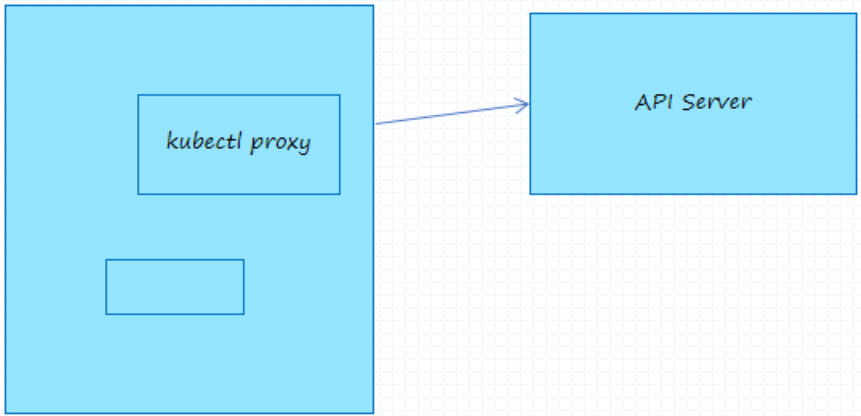

Deployment
Cronjob
DaemonSet
ReplicaSet
StatefulSet
Job
www.magedu.com
Pod
Container
Ingress
Service
Horizontal Pod AutoScaler
Pod Disruption Budget
Vertical Pod AutoScaler
Volume
ConfigMap
PersistentVolume Claim
Secret
Downward API

namespace

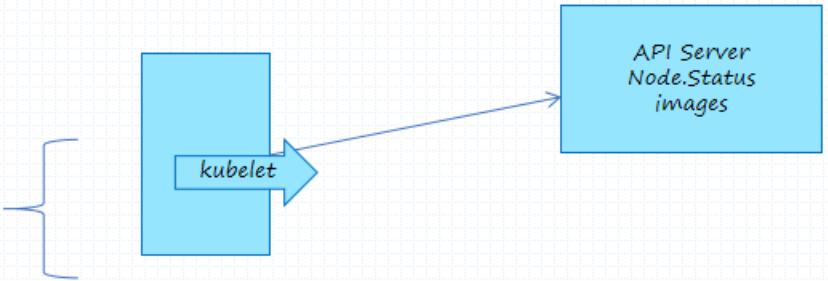
- 工作负载型（workload），Pod
  - stateful，stateless
  - Deployment，DaemonSet，StatefulSet->Operator
  - Job，Cronjob
- 服务发现和负载均衡
  - Service，Ingress
- 配置和存储
  - ConfigMap/Secret
  - PVC/PV
  - Downward API
  - role，rolebinding
- 集群级别的资源
  - namespace，node，clusterrole
- 元数据类型
  - LimitRange，...



- /apis/GROUP/VERSION/namespaces/NAMESPACE/pods/POD_NAME

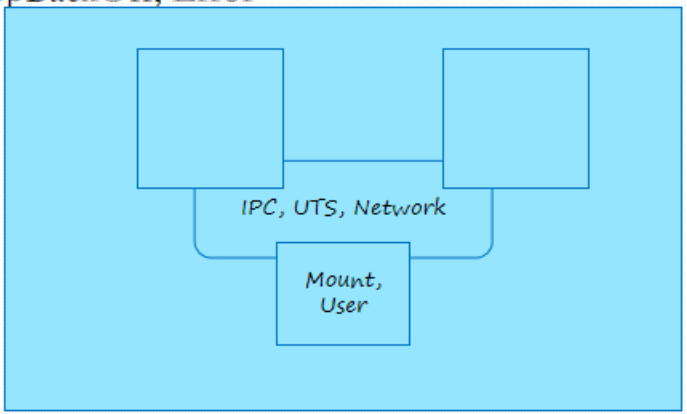


- 资源引用格式
  - <type>/<name>: pods/mypod, deployment/demoapp
  - <type> <name>: pods mypod, deployment demoapp
- lease, leases, kube-node-lease

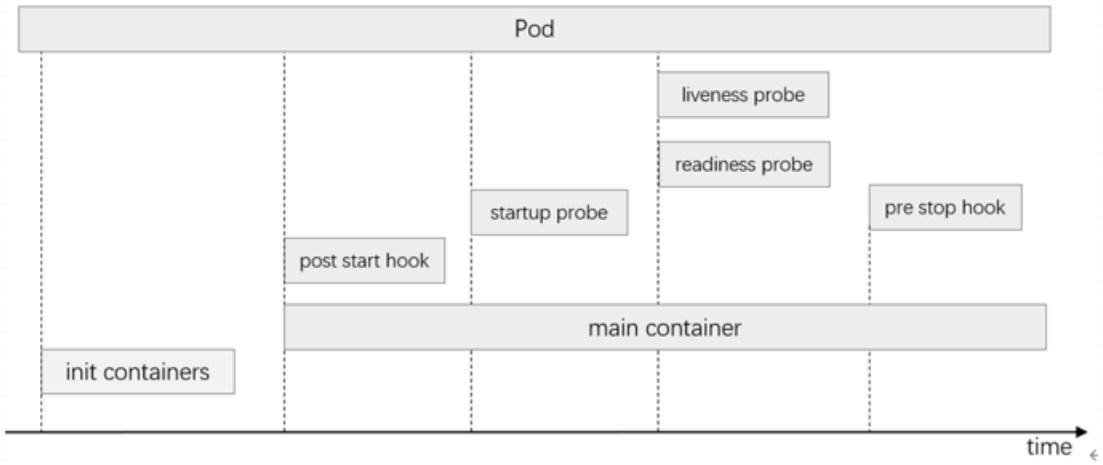


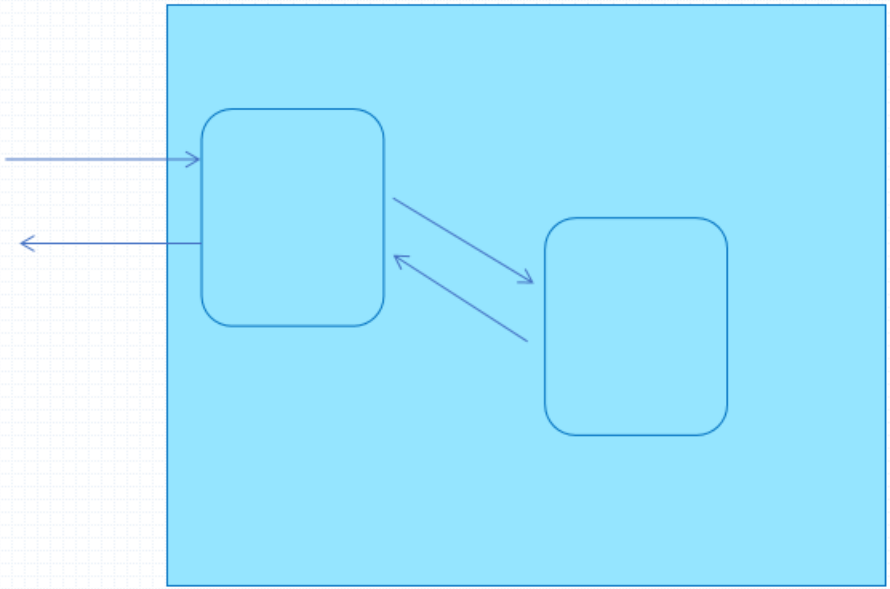
- 相位： Running, Pending(未调度成功，或仍处于下载镜像的过程中)
- Succeeded：成功终止， job,
- Failed
- Unknown：

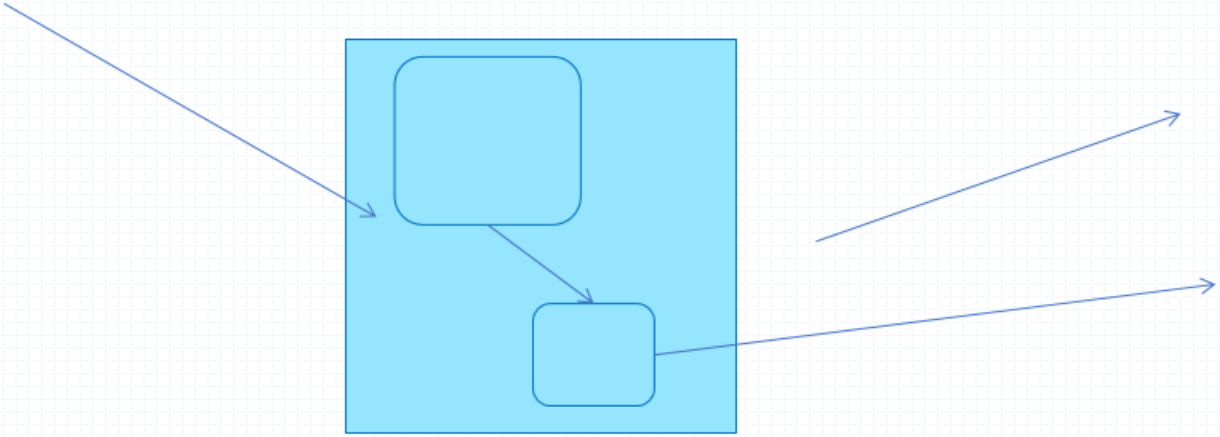
- 状态：CrashLoopBackOff, Error



- Pod中可以存在多个容器
  - Sidecar
  - Adapater
  - Ambassador

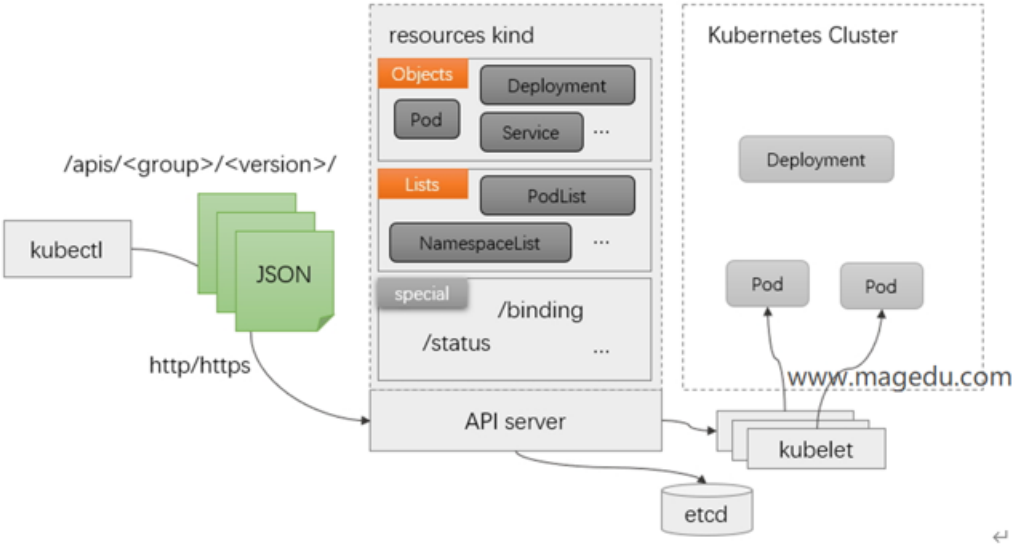


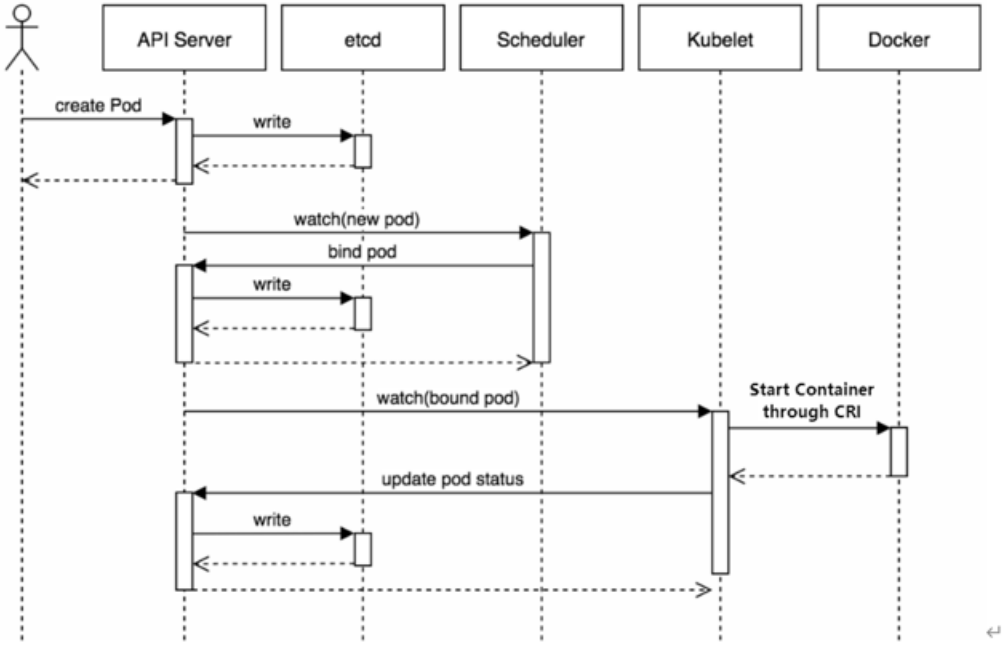




Security Context

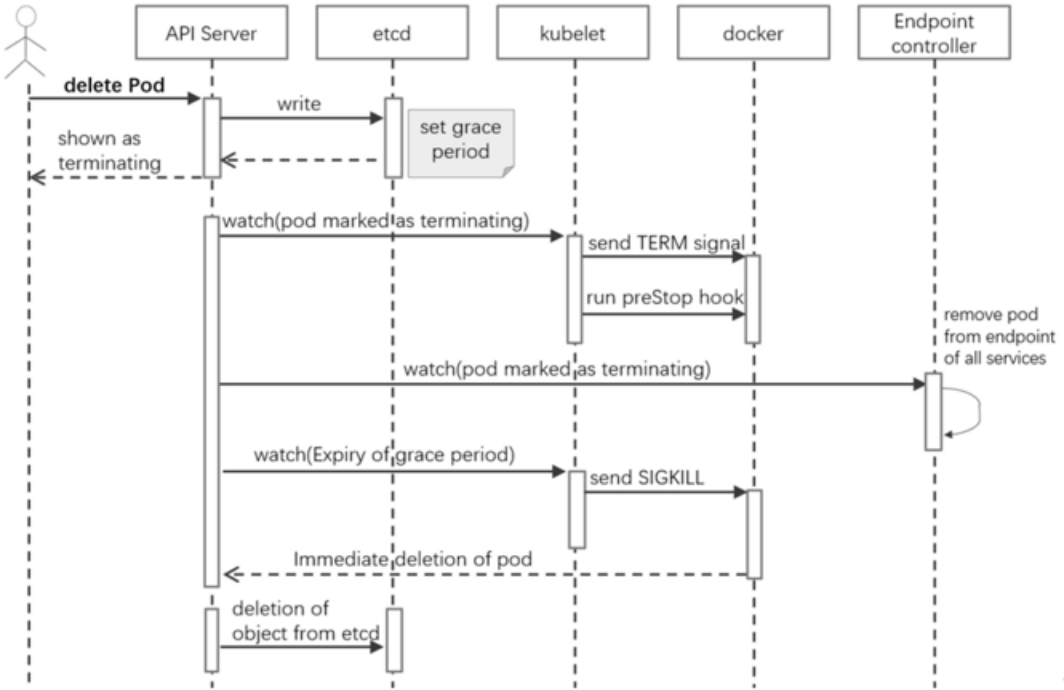
- 一组用于决定容器是如何创建和运行的约束条件，它们代表创建和运行容器时使用的运行时参数；
- Pod上的SC有两个级别
  - Pod级别
  - 容器级别
  - PSP：PodSecurityPolicy


resources kind
Objects
Deployment
Pod
Service
...
Lists
PodList
NamespaceList
...
special
/binding
/status
...
Kubernetes Cluster
Deployment
Pod
Pod
www.magedu.com
/apis/<group>/<version>/
kubectl
JSON
http/https
API server
kubelet
etcd



作业

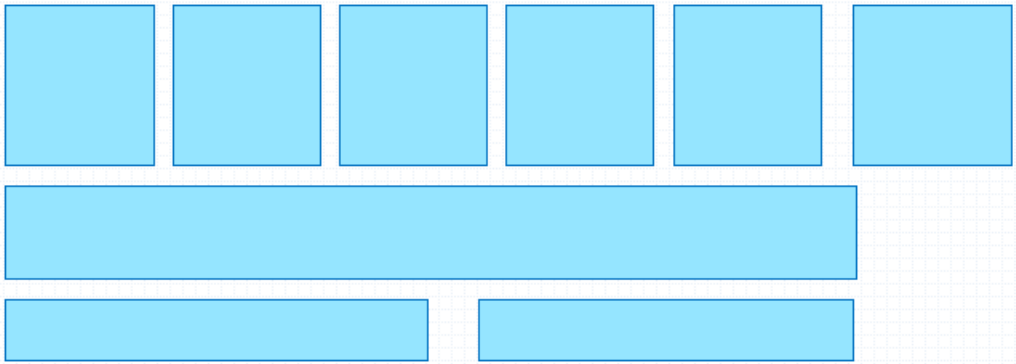
- 将MySQL运行为Pod；
- 将Wordpress运行为Pod，且能将数据存储于MySQL Pod中；
- 将MySQL和Wordpress运行于test名称空间中；
- 让Wordpress能在k8s集群之外进行访问；



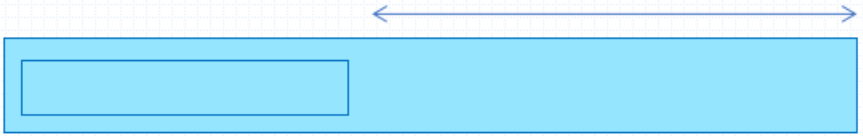
Cloud Native

HTTP/HTTPS，/URI
VirtualHost
埋点，探针

resource, request, limit



◆退避算法，1,0； 2,10; 3,20; 4,40;, 5,80;6, 160, 7, 300; CrashLoopBackOff



计算资源

- CPU， Memory
- CPU，可压缩型，
- Memory，不可压缩型，2G，1G -> 2G, OOM Killer

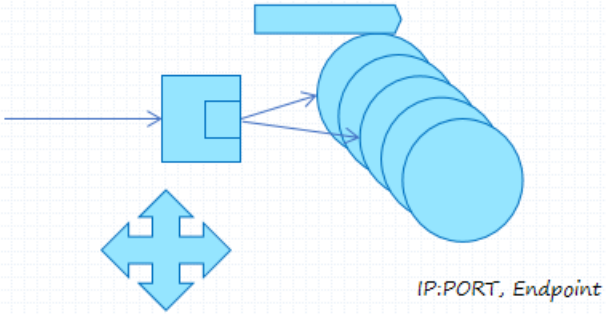
- CPU单位：多少CPU核心，1核=1000m核， 1000m
- Memory: 1Ki, 1Mi

- k8s
  - Pod, Service, Volume,

- request：确保节点至少为Pod或容器配备的资源最小量；

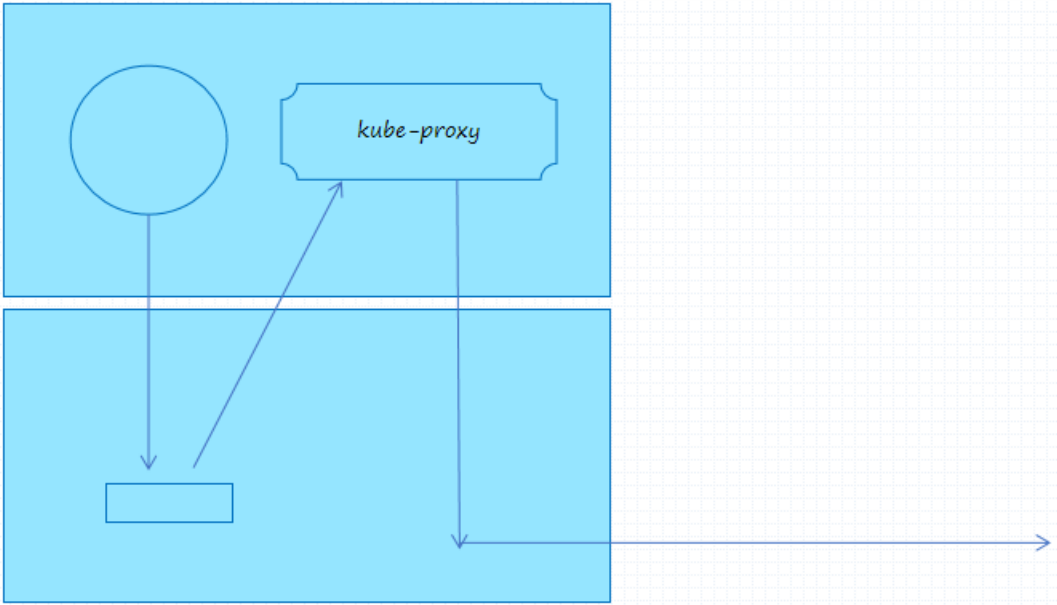
label selector, label

- Service：标准资源类型， Service Controller
  - 为动态的一组Pod提供一个固定的访问入口，ClusterIP（Cluster Network）
- Endpoint：标准资源类型, Endpoint Controller



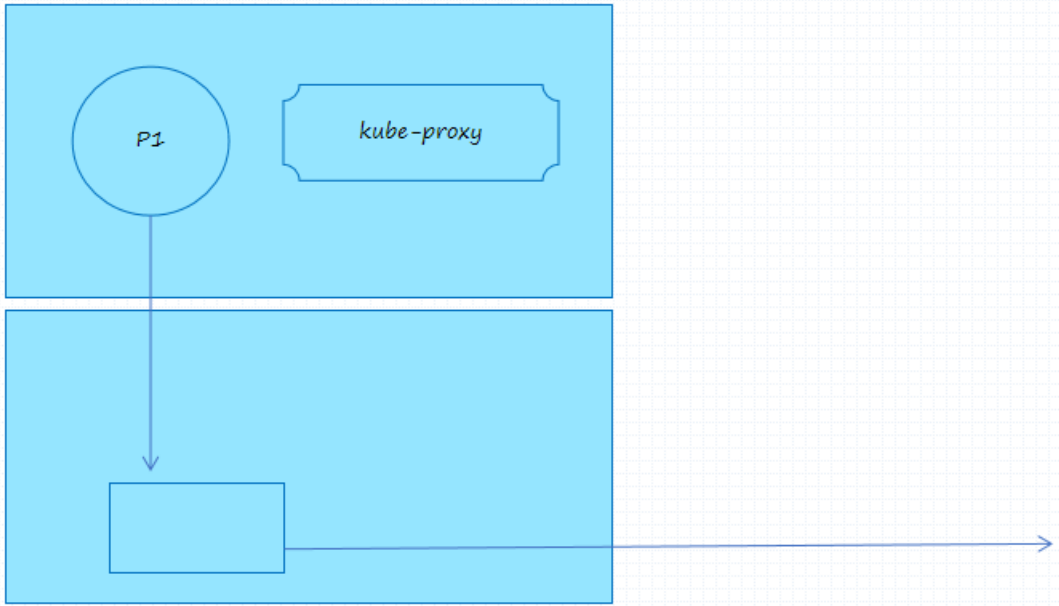
10.96.0.0/12网络中动态选择一个；

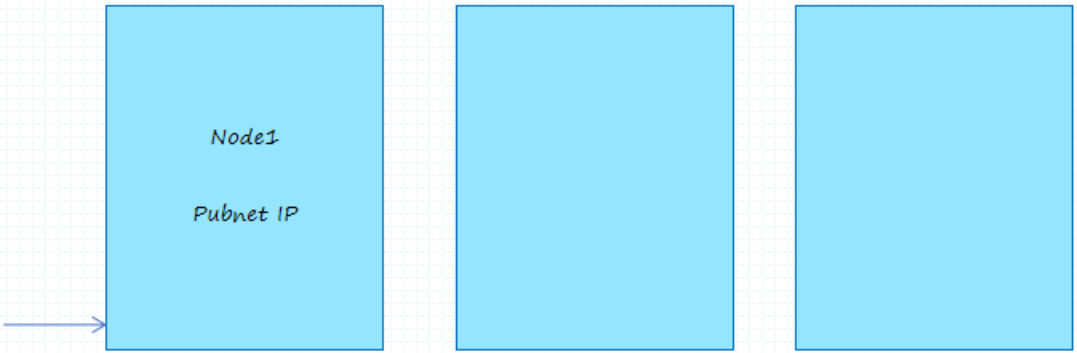
- kube-proxy， Service Controller位于各节点上agent；
  - 代理模型
  - Userspace模型：Pod -> Service，iptables拦截规则，但自己不做调度

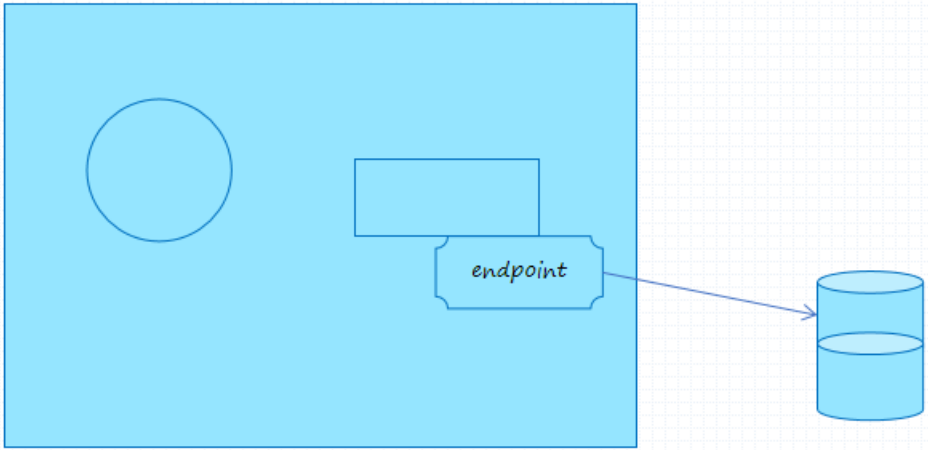


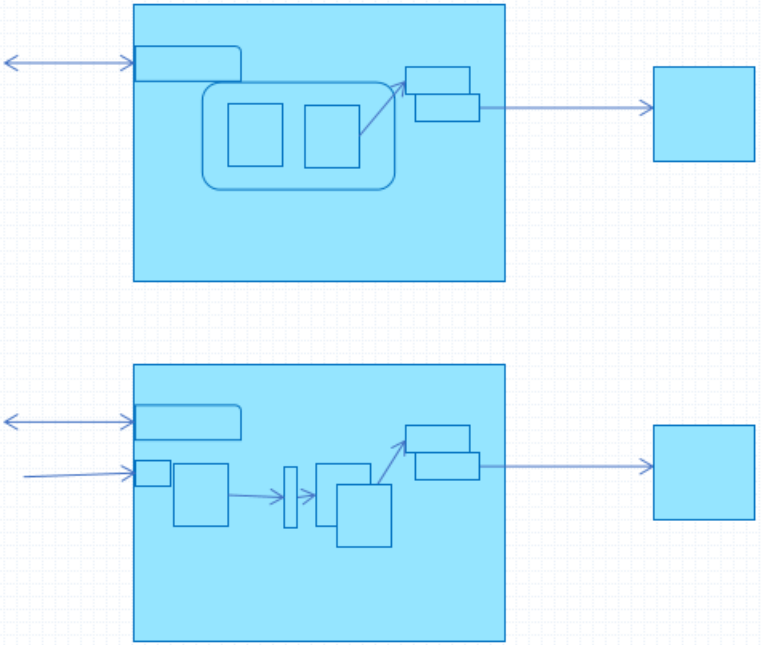
- iptables模式， 一个service会生成大量的规则；
- ipvs模式

- 不变IP， ClusterIP

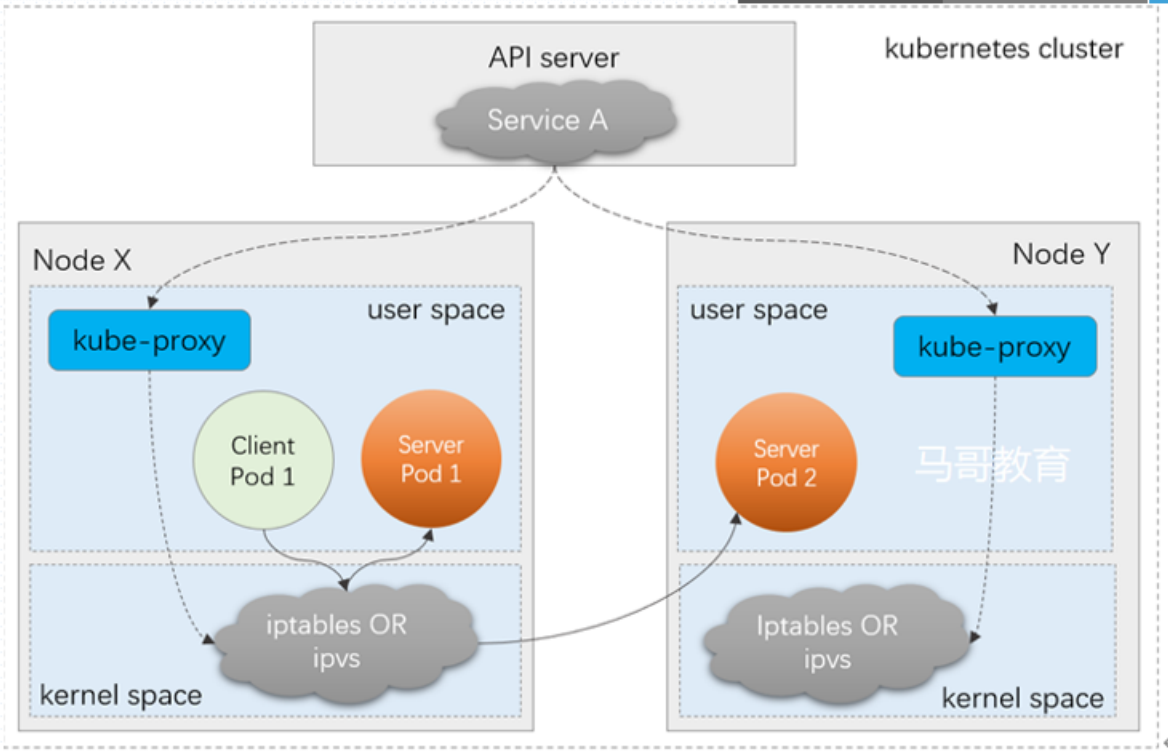


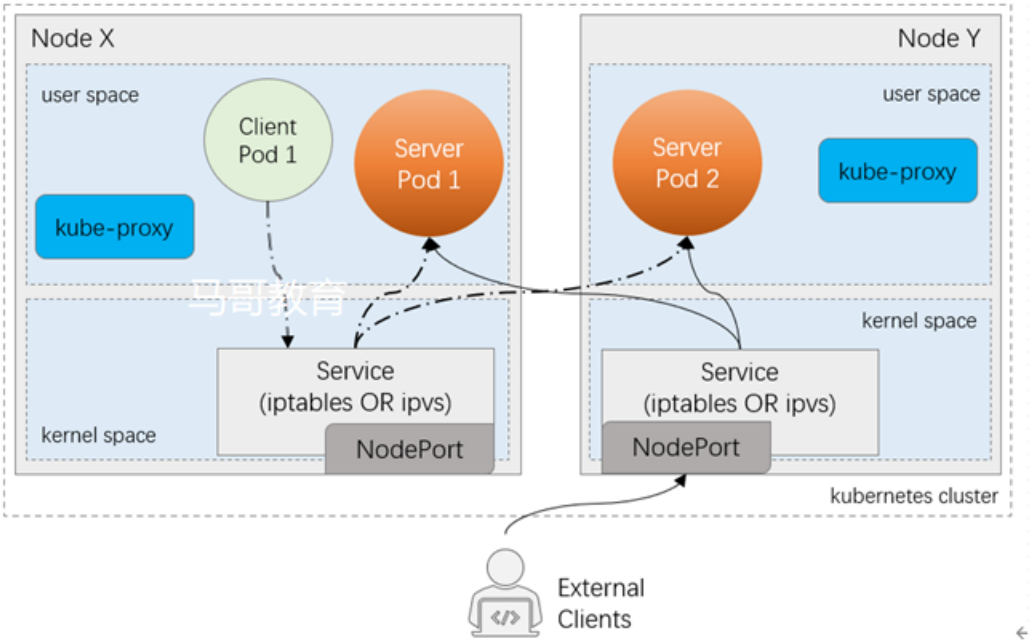

Node1
Pubnet IP

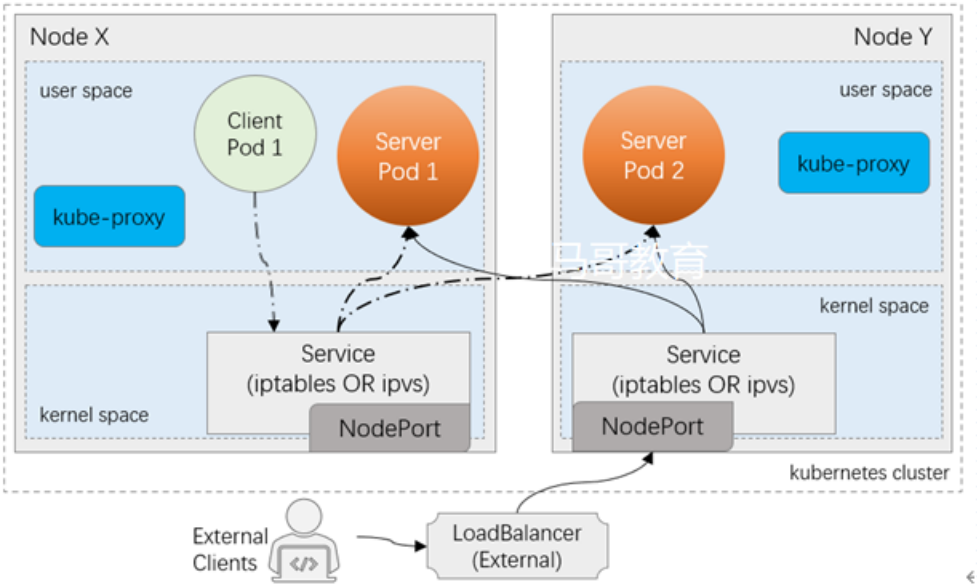

endpoint

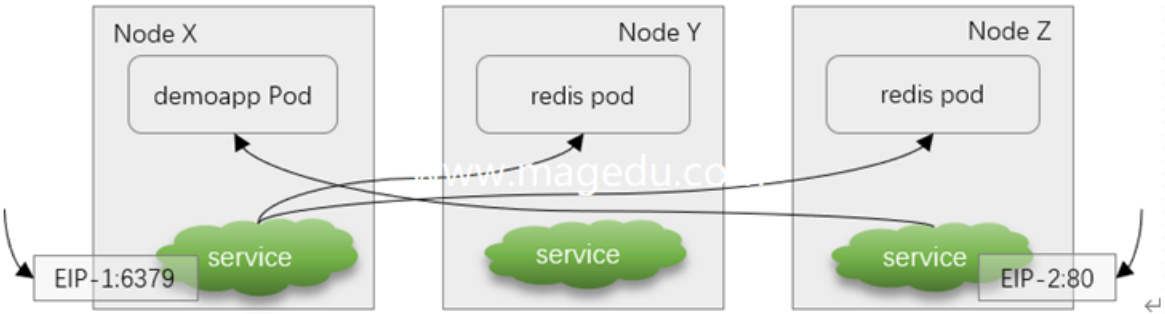


kube-proxy和service

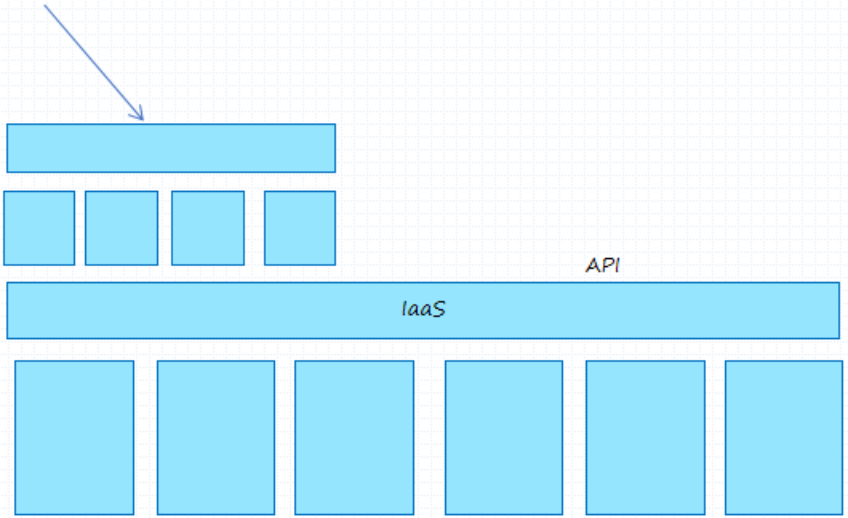


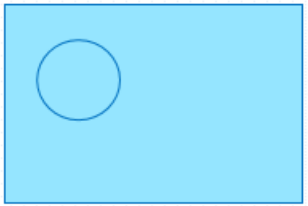

Node X
user space
Client Pod 1
Server Pod 1
kube-proxy
kernel space
Service (iptables OR ipvs)
NodePort
Node Y
user space
Server Pod 2
kube-proxy
kernel space
Service (iptables OR ipvs)
NodePort
kubernetes cluster
External Clients

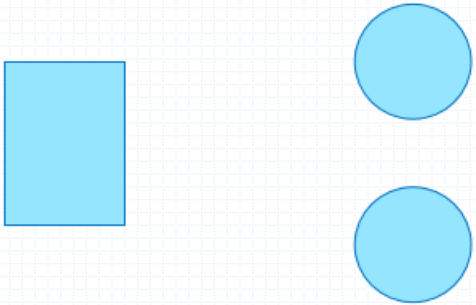

Node X
user space
Client Pod 1
Server Pod 1
kube-proxy
kernel space
Service (iptables OR ipvs)
NodePort
Node Y
user space
Server Pod 2
kube-proxy
kernel space
Service (iptables OR ipvs)
NodePort
kubernetes cluster
External Clients
LoadBalancer (External)

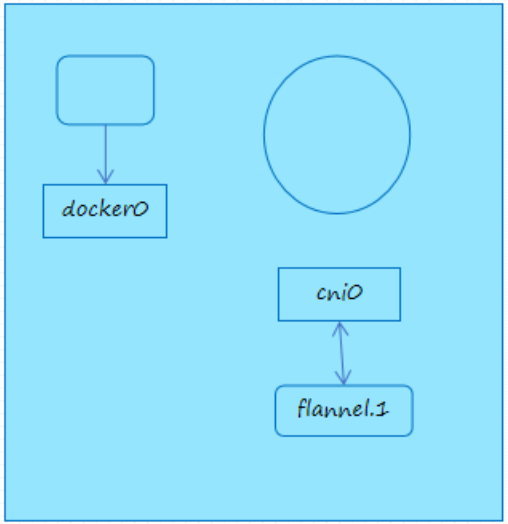

Node X
demoapp Pod
Node Y
redis pod
Node Z
redis pod
service
service
service
EIP-1:6379
EIP-2:80

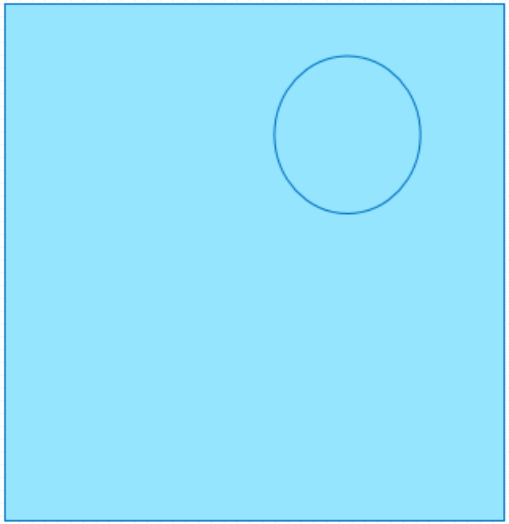
- Service, Endpoints
  - Service：Pod应用，提供固定访问端点，ClusterIP，Service Name
  - Endpoint：
- Service：
  - Userspace：kube-proxy代理服务进程
  - iptables：
  - ipvs：
- Service 类型
  - ClusterIP：流量的惟一入口；ExternalIP
  - NodePort：NodeIP:NodePort，是流量的第二入口；
  - LoadBalancer：为各节点上的NodePort提供一个外部负载均衡器；
  - ExternalName：



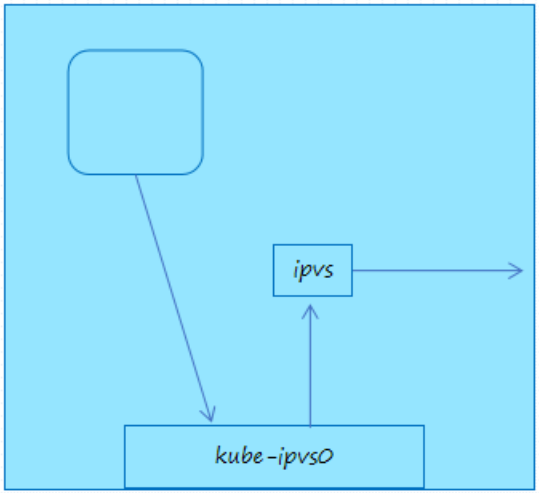




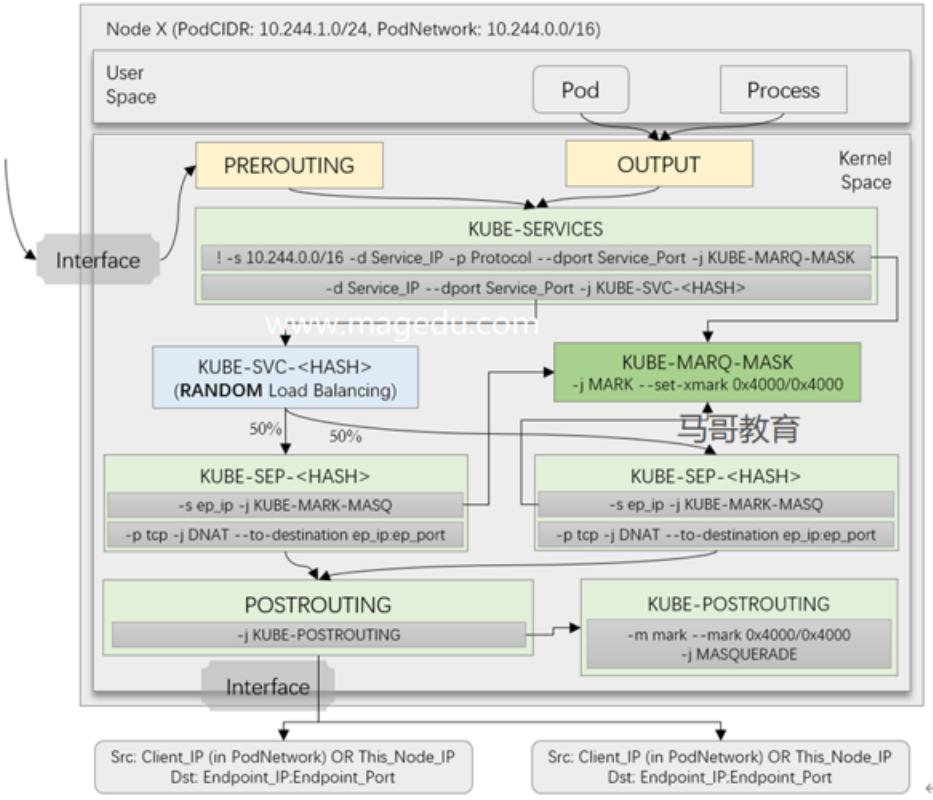

docker0
cni0
flannel.1

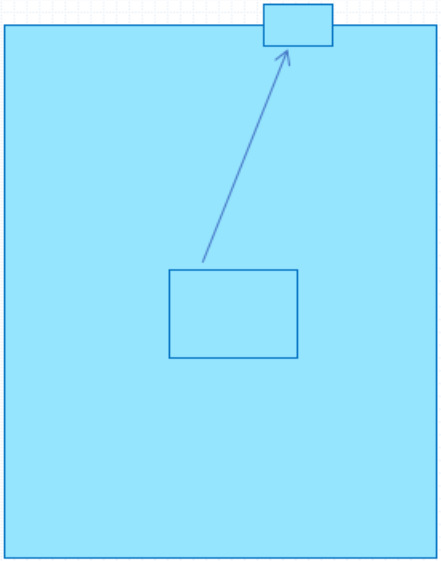


iptables规则

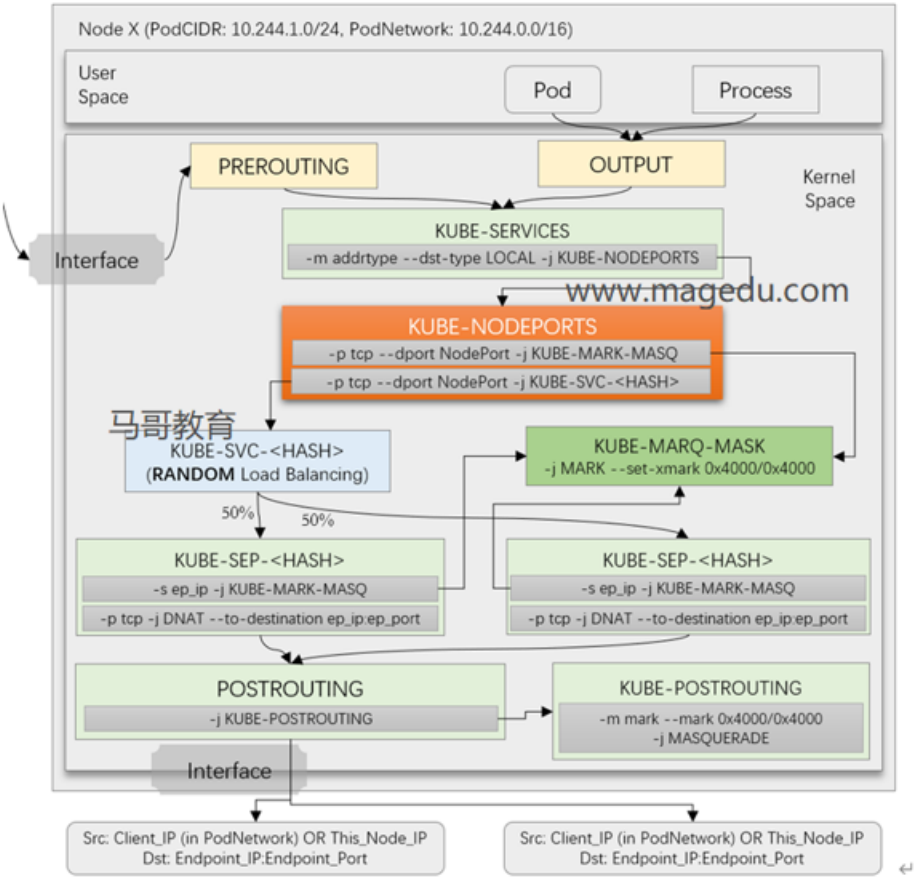


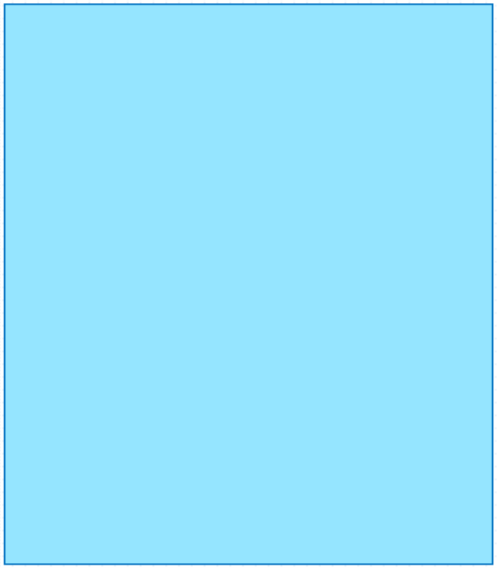
ClusterIP

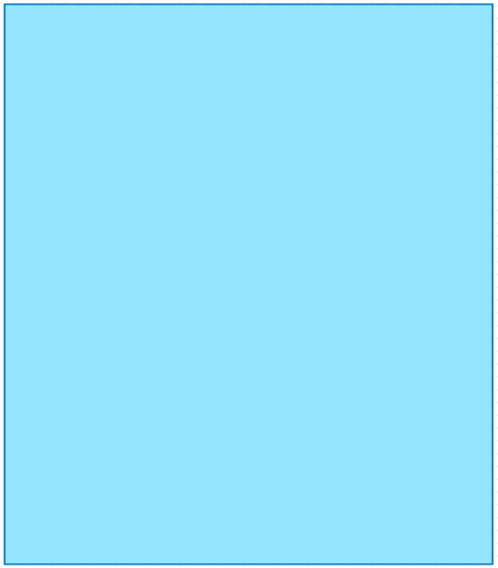

Node X (PodCIDR: 10.244.1.0/24, PodNetwork: 10.244.0.0/16)
User Space
Pod
Process
Kernel Space
PREROUTING
OUTPUT
Interface
KUBE-SERVICES
! -s 10.244.0.0/16 -d Service_IP -p Protocol --dport Service_Port -j KUBE-MARQ-MASK
-d Service_IP --dport Service_Port -j KUBE-SVC-<HASH>
KUBE-SVC-<HASH>
(RANDOM Load Balancing)
KUBE-MARQ-MASK
-j MARK --set-xmark 0x4000/0x4000
50%
50%
KUBE-SEP-<HASH>
-s ep_ip -j KUBE-MARK-MASQ
-p tcp -j DNAT --to-destination ep_ip:ep_port
KUBE-SEP-<HASH>
-s ep_ip -j KUBE-MARK-MASQ
-p tcp -j DNAT --to-destination ep_ip:ep_port
POSTROUTING
-j KUBE-POSTROUTING
KUBE-POSTROUTING
-m mark --mark 0x4000/0x4000
-j MASQUERADE
Interface
Src: Client_IP (in PodNetwork) OR This_Node_IP
Dst: Endpoint_IP:Endpoint_Port
Src: Client_IP (in PodNetwork) OR This_Node_IP
Dst: Endpoint_IP:Endpoint_Port



NodePort

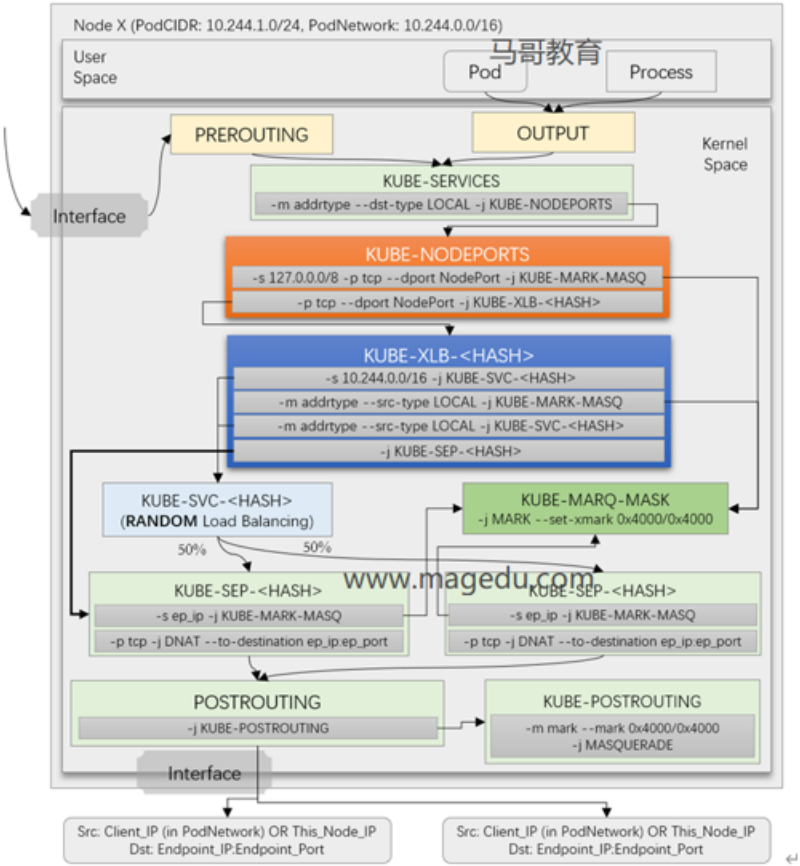

Node X (PodCIDR: 10.244.1.0/24, PodNetwork: 10.244.0.0/16)
User Space
Pod
Process
Kernel Space
PREROUTING
OUTPUT
Interface
KUBE-SERVICES
-m addrtype --dst-type LOCAL -j KUBE-NODEPORTS
www.magedu.com
KUBE-NODEPORTS
-p tcp --dport NodePort -j KUBE-MARK-MASQ
-p tcp --dport NodePort -j KUBE-SVC-<HASH>
马哥教育
KUBE-SVC-<HASH>
(RANDOM Load Balancing)
KUBE-MARQ-MASK
-j MARK --set-xmark 0x4000/0x4000
50%
50%
KUBE-SEP-<HASH>
-s ep_ip -j KUBE-MARK-MASQ
-p tcp -j DNAT --to-destination ep_ip:ep_port
KUBE-SEP-<HASH>
-s ep_ip -j KUBE-MARK-MASQ
-p tcp -j DNAT --to-destination ep_ip:ep_port
POSTROUTING
-j KUBE-POSTROUTING
KUBE-POSTROUTING
-m mark --mark 0x4000/0x4000
-j MASQUERADE
Interface
Src: Client_IP (in PodNetwork) OR This_Node_IP
Dst: Endpoint_IP:Endpoint_Port
Src: Client_IP (in PodNetwork) OR This_Node_IP
Dst: Endpoint_IP:Endpoint_Port



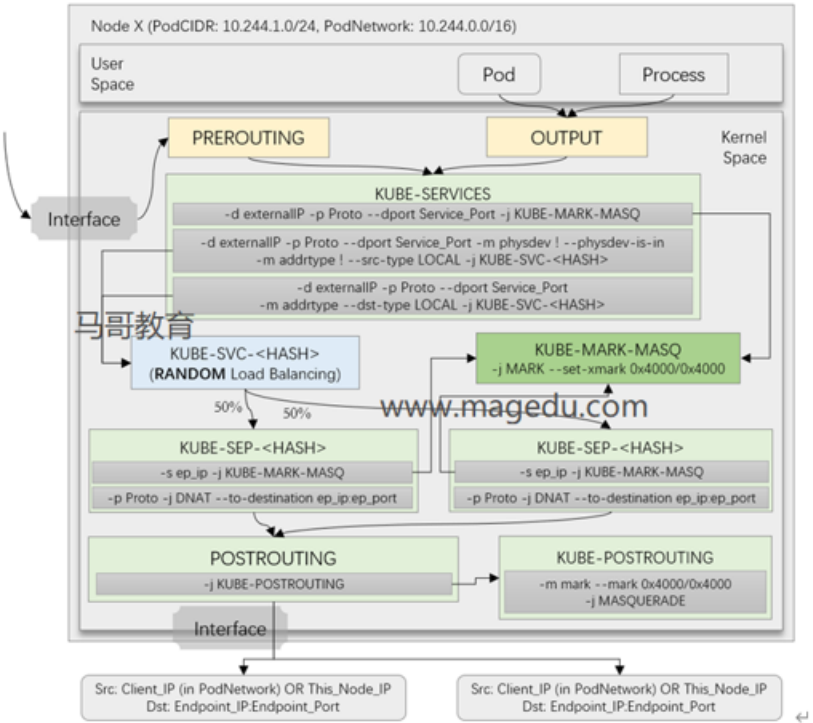


NodePort

- Local流量策略

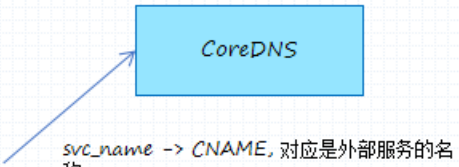


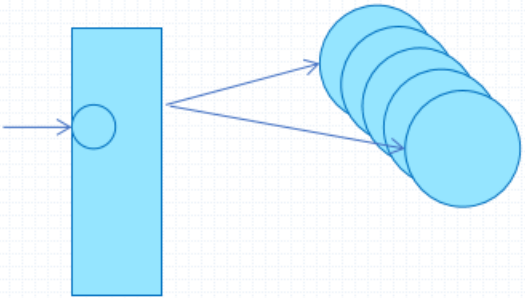
externalIP



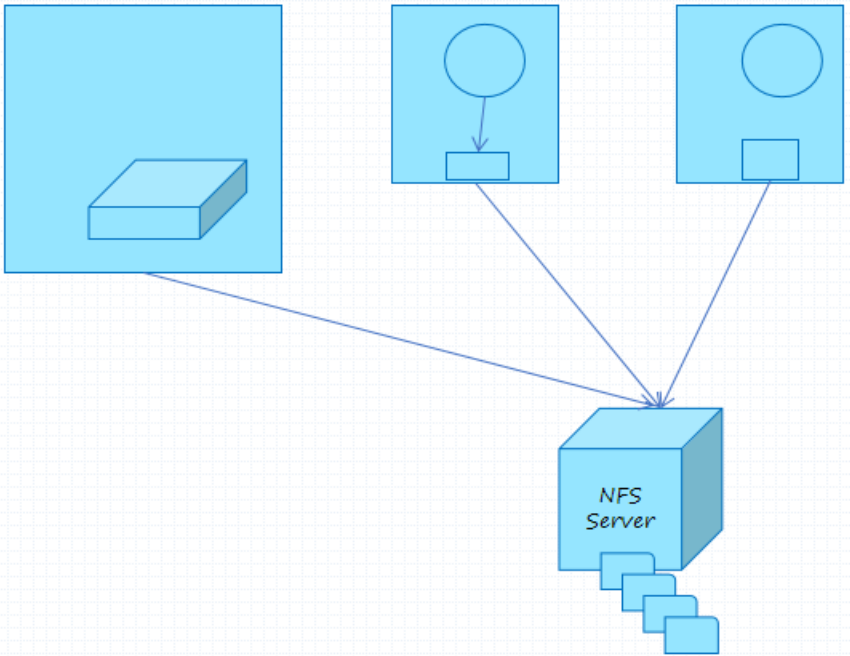
- Service
  - iptables代理模式
  - ipvs代理模式：kube-ipvs0，将所有的ClusterIP绑定在该接口； 而后将每个Service定义为虚拟服务器；
- 服务发现：
  - 环境变量：Kubernetes Service环境变量，Docker Link环境变量；
  - DNS服务：API Server
    - 对于每个Service
      - ✓ A或AAAA记录； service name -> service ip
      - ✓ server name: <svc_name>.<namespace>.svc.<zone>, cluster.local. ， 节点域名；
      - ✓ SRV：端口解析
      - ✓ PTR：service ip -> service name
  - Pod，/etc/resolve.conf
    - namserver 10.96.0.10
    - kube-system， kubedns

svc_name -> clusterip, StatefulSet

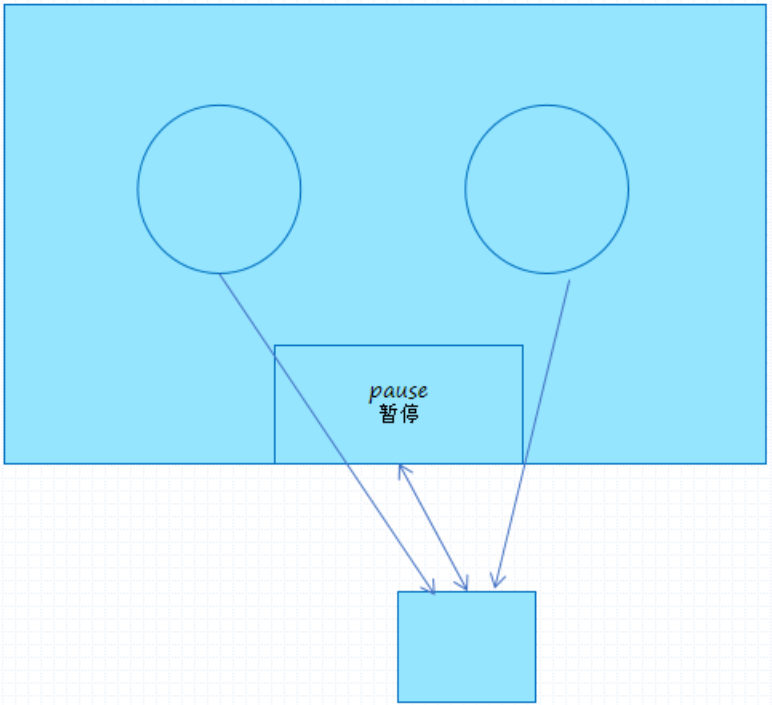


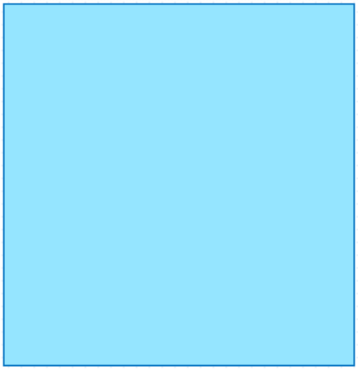


- Docker, 文件系统，与Docker容器具有同样的生命周期；
- 外部存储空间：
  - Host：
  - Network Storage

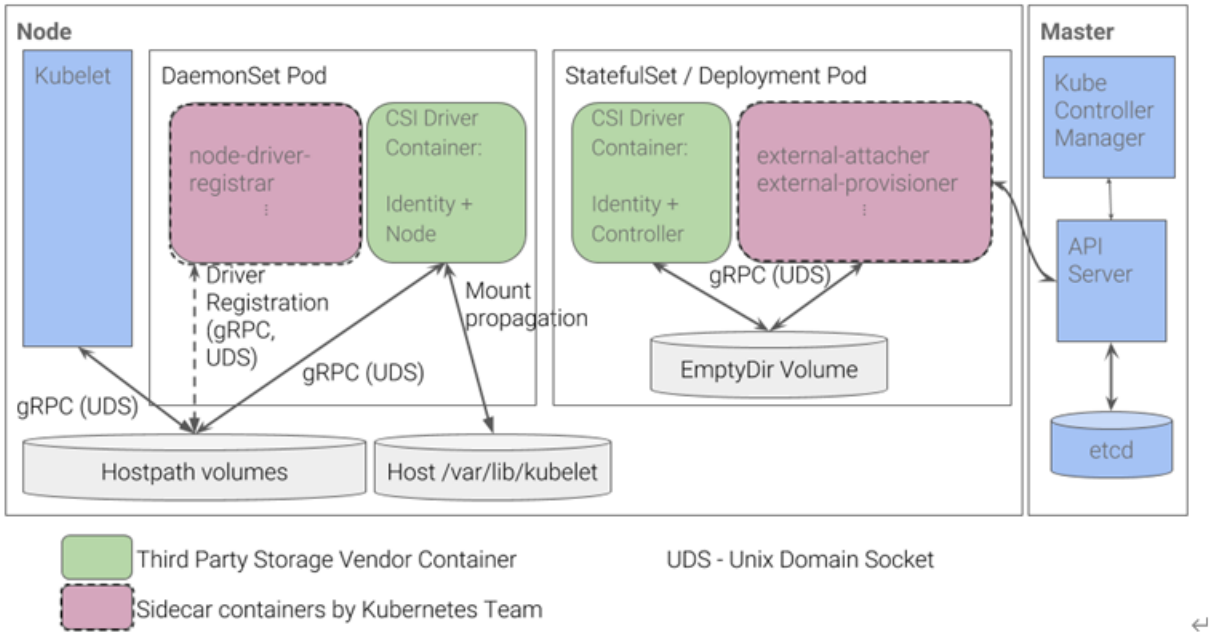


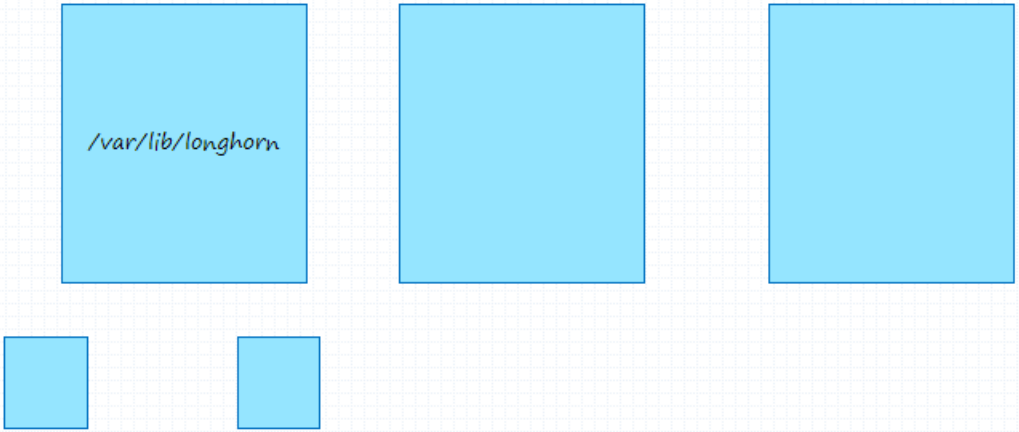
- kubelet --> Container Engine
  - 管理存储卷的功能，也是kubelet
  - Volume Plugin：卷插件

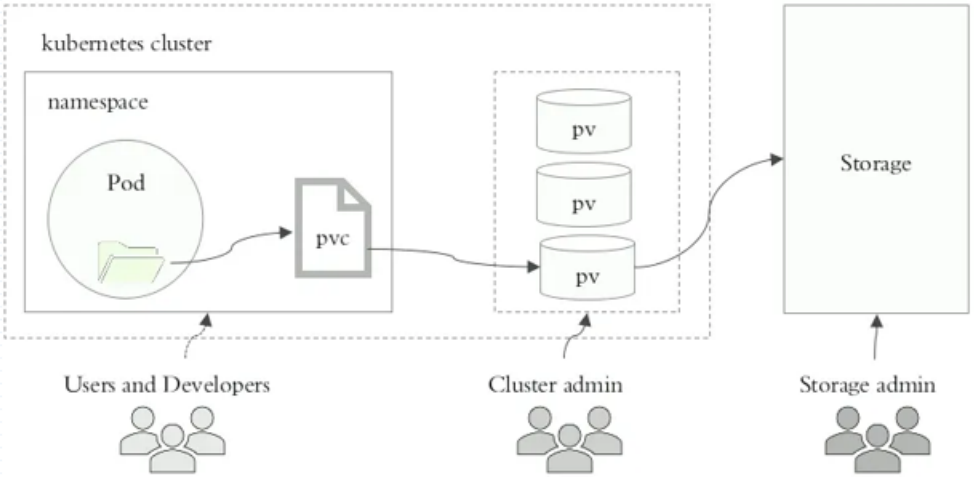

pause
暂停

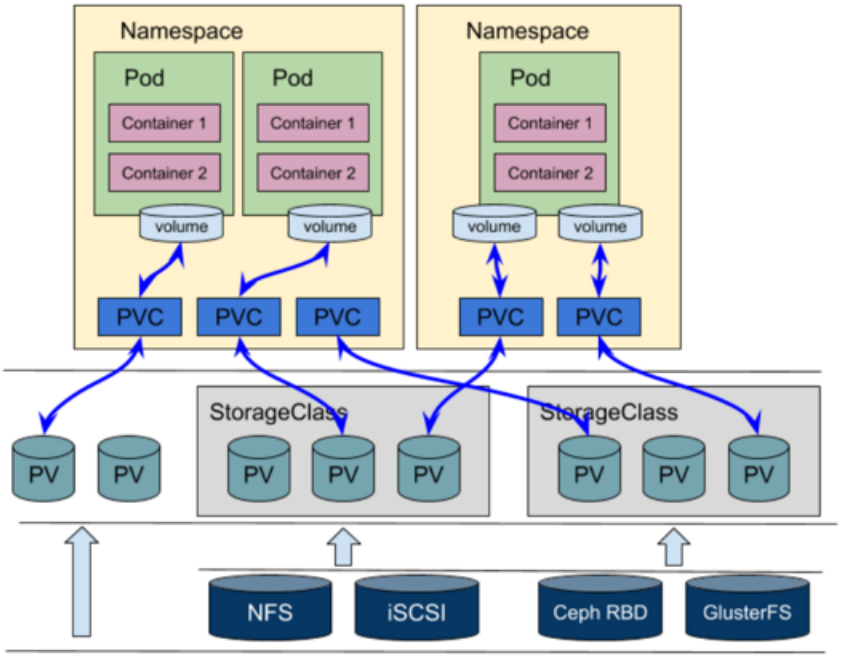


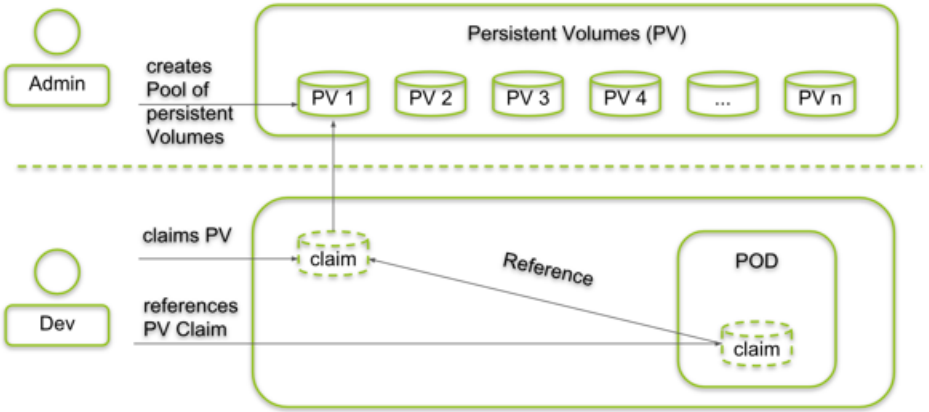
◆存储卷，pv, PVC

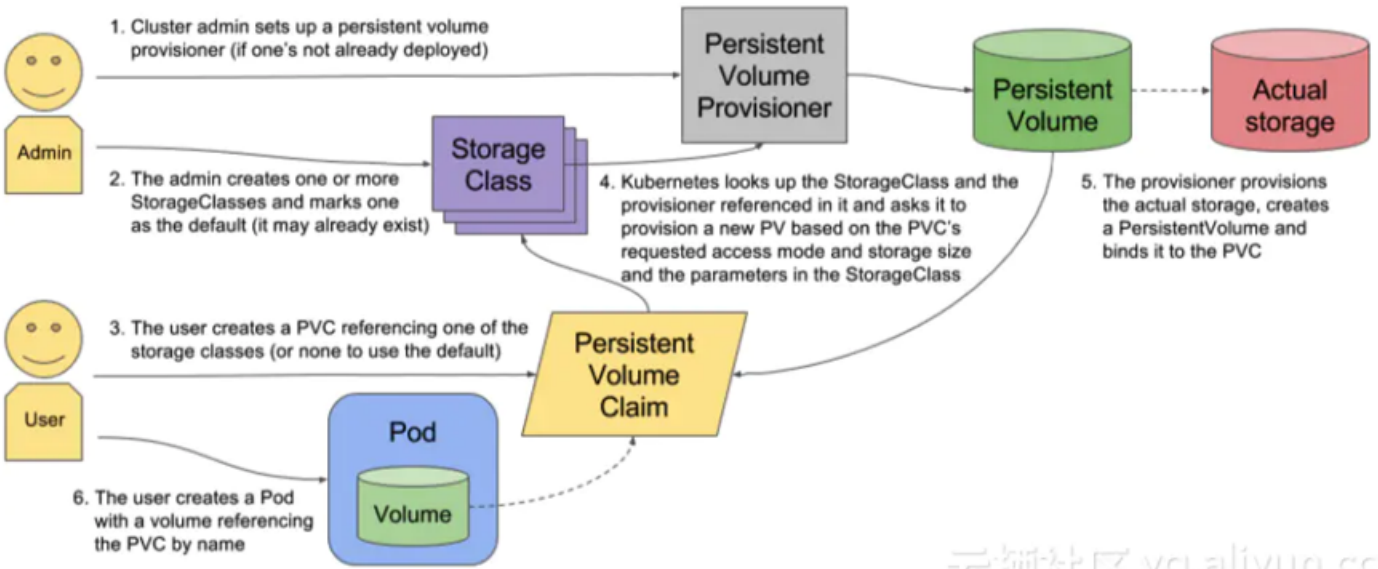

Node
Kubelet
DaemonSet Pod
node-driver-registrar
CSI Driver Container:
Identity + Node
Driver Registration (gRPC, UDS)
gRPC (UDS)
Mount propagation
gRPC (UDS)
Hostpath volumes
Host /var/lib/kubelet
StatefulSet / Deployment Pod
CSI Driver Container:
Identity + Controller
external-attacher
external-provisioner
gRPC (UDS)
EmptyDir Volume
Master
Kube Controller Manager
API Server
etcd
Third Party Storage Vendor Container
Sidecar containers by Kubernetes Team
UDS - Unix Domain Socket

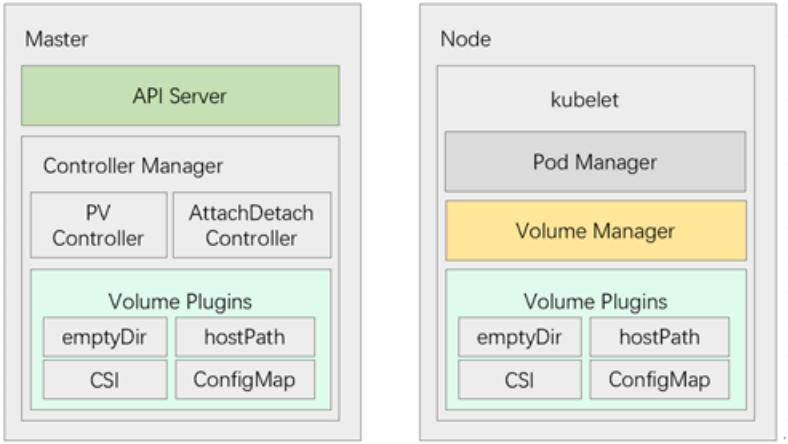

/var/lib/longhorn

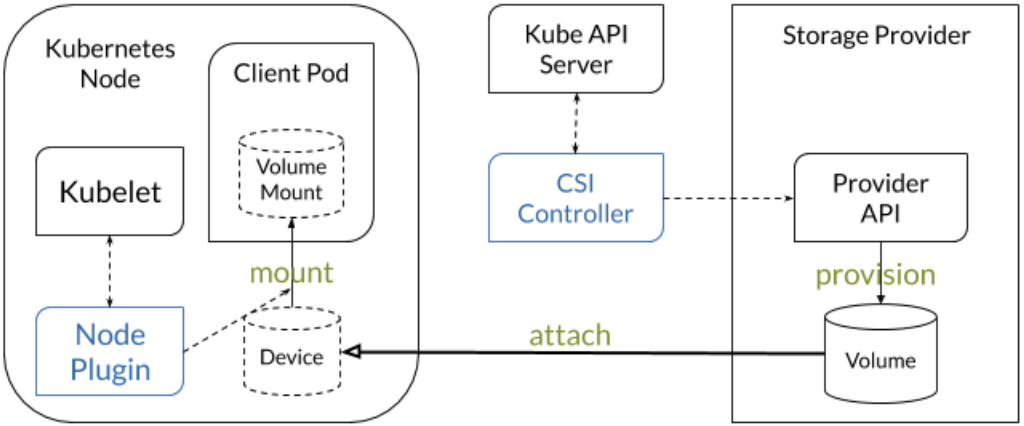


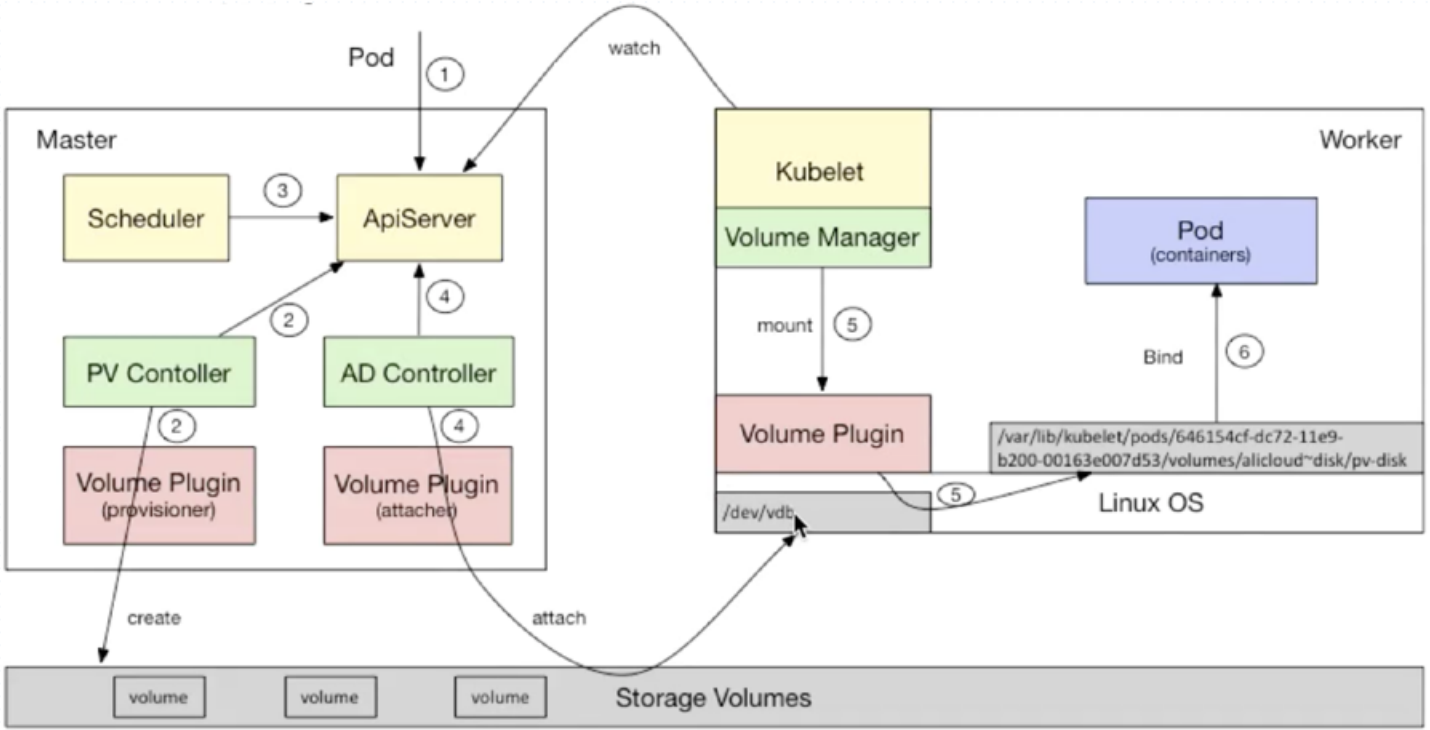

Namespace
Pod
Container 1
Container 2
volume
Pod
Container 1
Container 2
volume
Namespace
Pod
Container 1
Container 2
volume
volume
PVC
PVC
PVC
PVC
PVC
StorageClass
StorageClass
PV
PV
PV
PV
PV
PV
PV
PV
NFS
iSCSI
Ceph RBD
GlusterFS


Admin
creates
Pool of
persistent
Volumes
Persistent Volumes (PV)
PV 1
PV 2
PV 3
PV 4
...
PV n
claims PV
claim
Reference
POD
Dev
references
PV Claim
claim

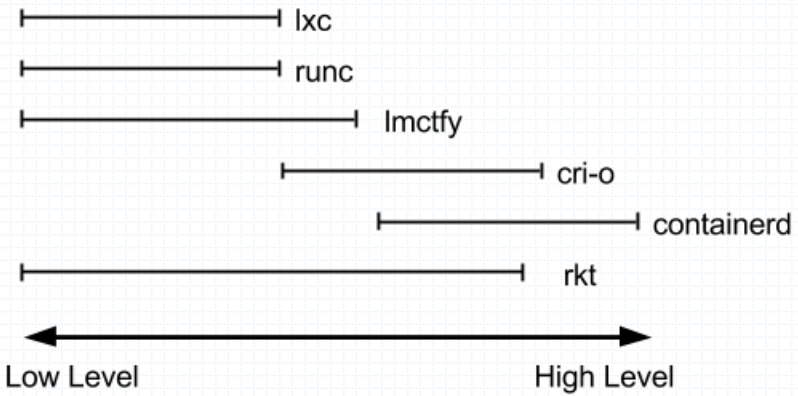








OCI(Open Container Initiative), LSF

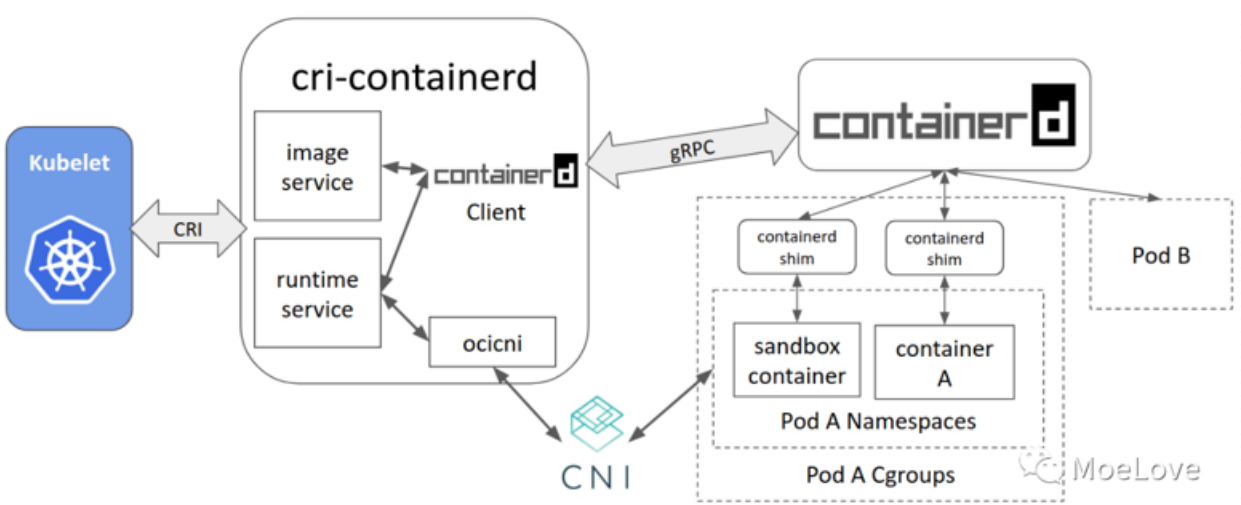


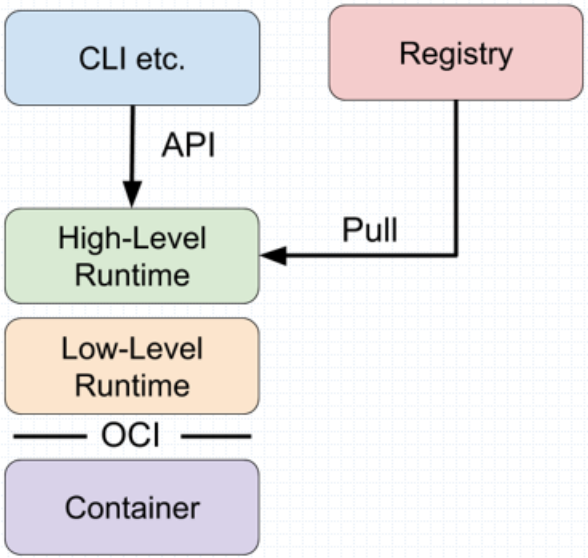
OCI

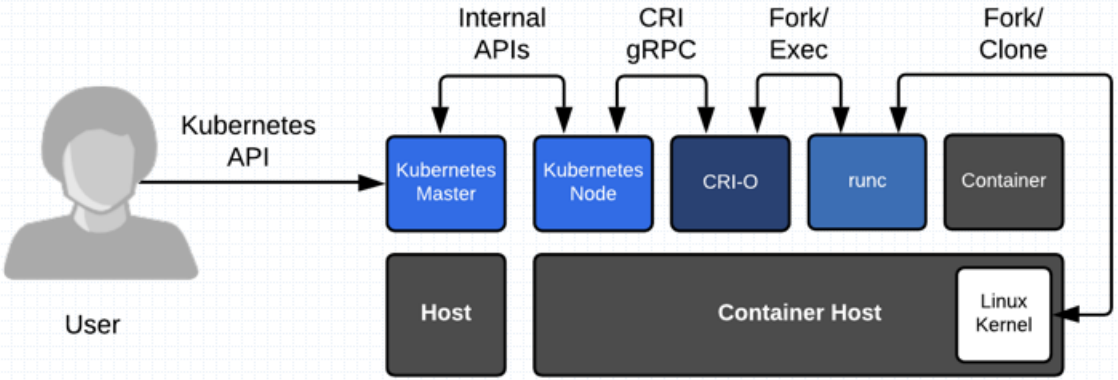
- runtime-spec，容器运行时 规范
- image-spec，镜像 规范

- 容器镜像格式， runtime（runC), 捐给了OCI

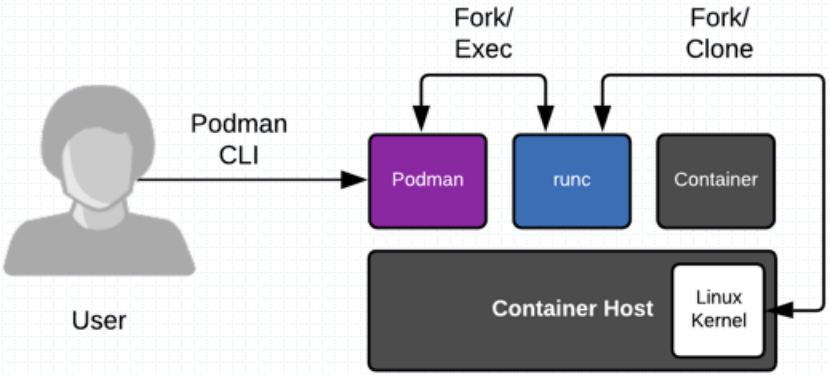
- 2017， v1.11，containerd捐给了CNCF



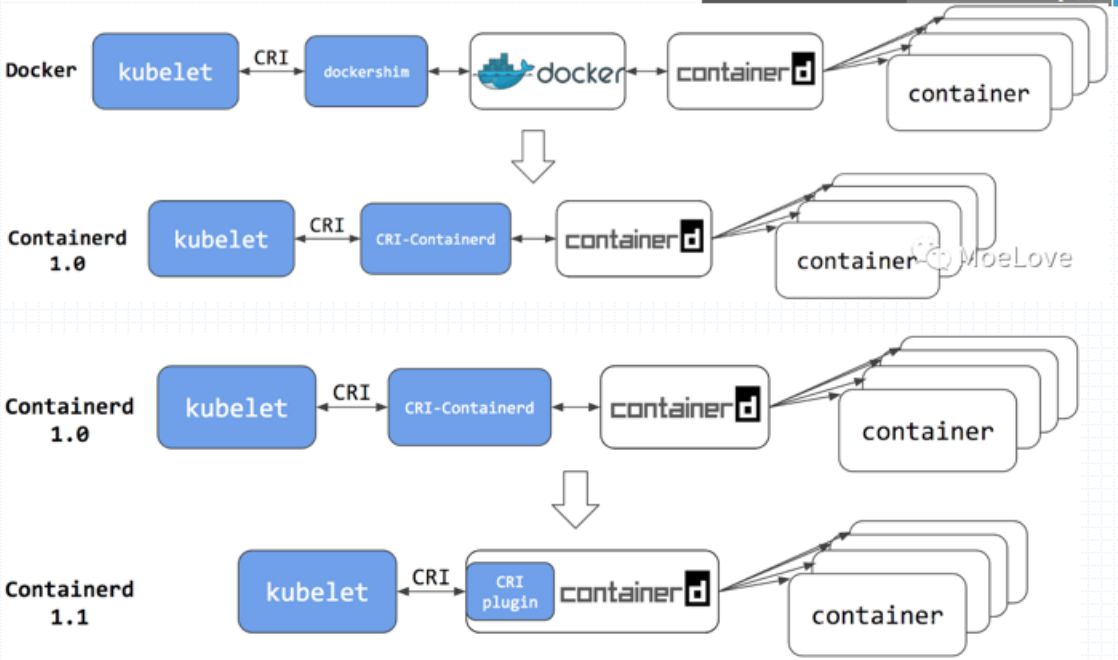


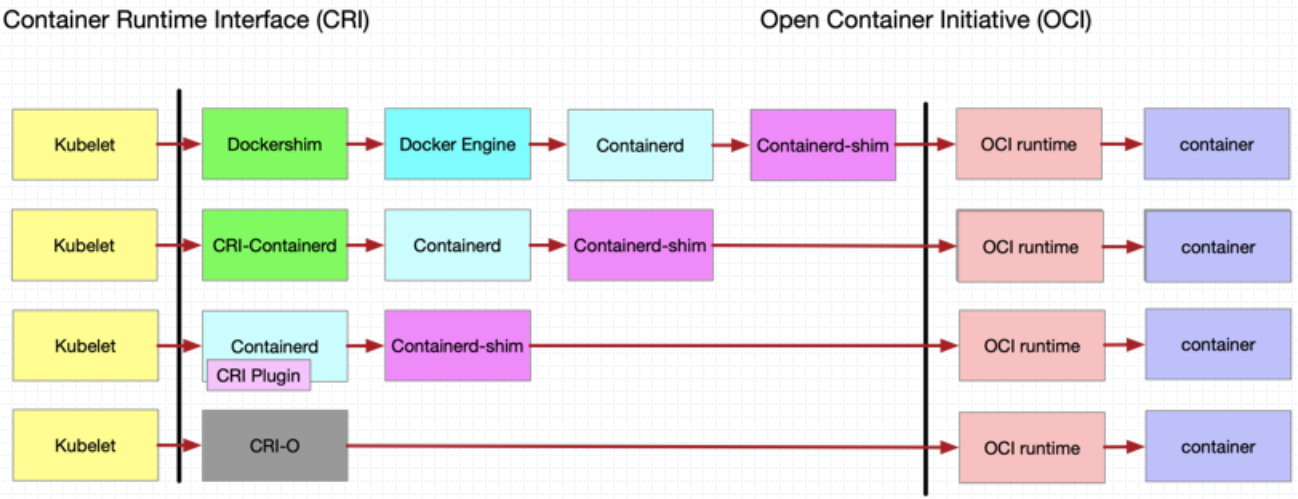


How containers run in a Kubernetes cluster



How containers run with a container engine





CentOS 8明年结束使命 开发重点将转移到CentOS Stream上

原创 cnBeta 2020-12-08 22:50:04

对于那些想要一个免费的、几乎相同的CentOS 8提供的当前RHEL8构建版本的人来说，一个足以令他们非常失望的消息传来，这一免费午餐将在2021年结束。**CentOS 8将在2021年结束支持，往后CentOS 7作为长期支持版本将继续被支持直到其生命周期结束，CentOS Stream将作为工作重点消息最近成为焦点。**

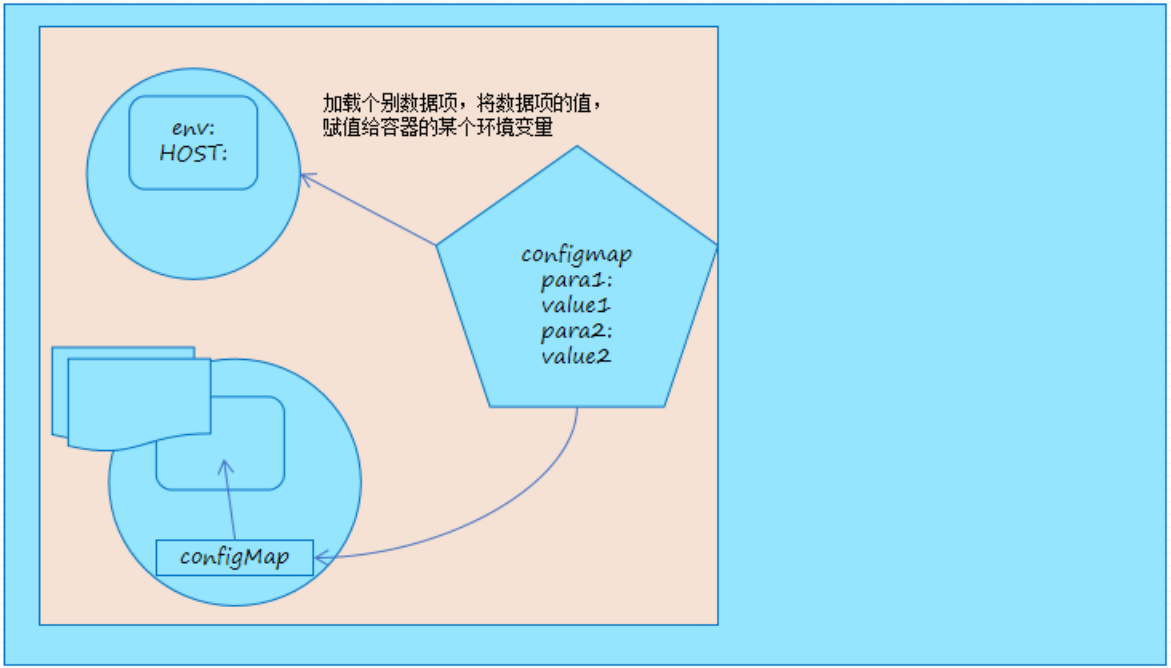
在2021年底，CentOS 8将不再维护，但CentOS 7将以支持维护的状态坚持到2024年。

CentOS 8.2

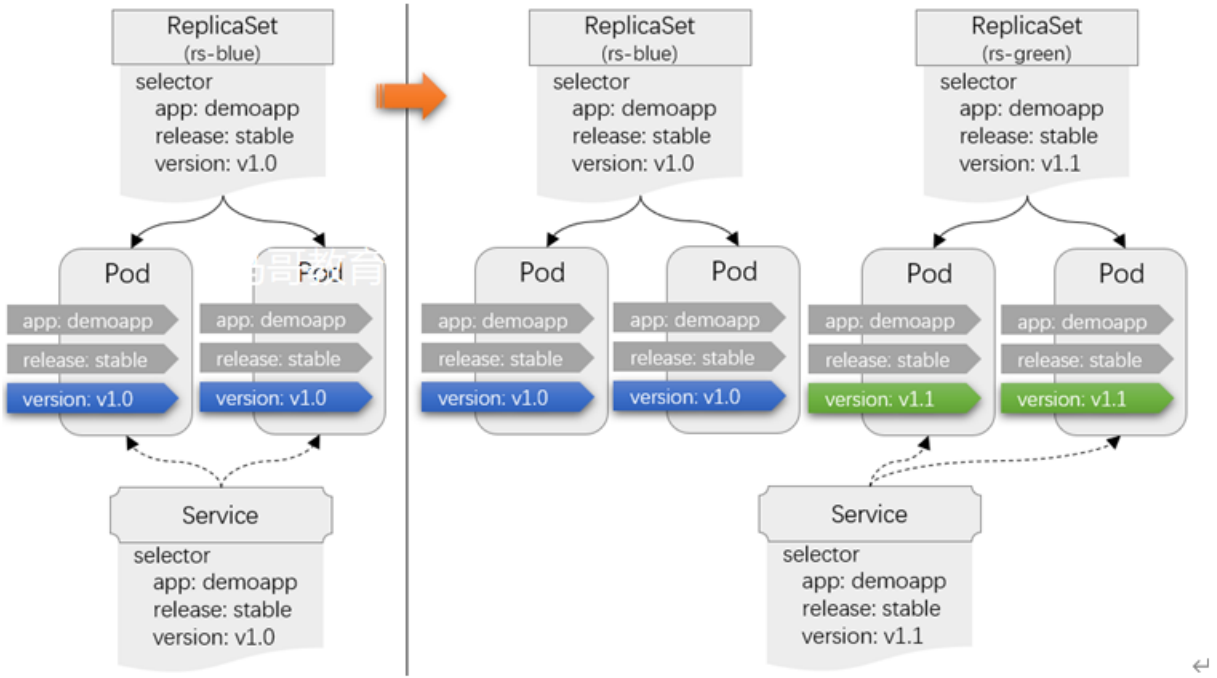
The Community ENTerprise Operating System

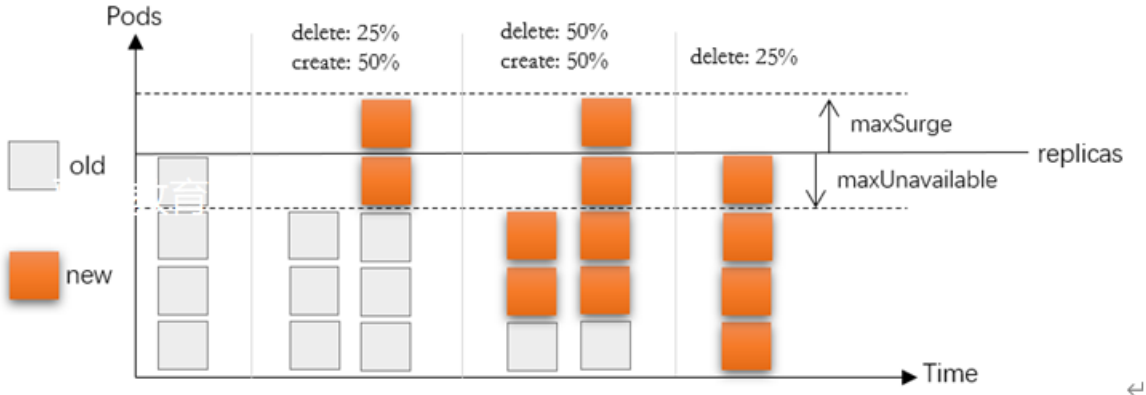
CentOS项目今天通过CentOS博客宣布了这一重点的转变。CentOS项目今后将只专注于CentOS Stream，作为红帽企业Linux的上游/开发分支。发行方鼓励CentOS 8用户开始向CentOS Stream 8过渡。同时，红帽还表示，英特尔将在CentOS Stream上与他们和社区合作。红帽还谈到Facebook在他们的数据中心使用CentOS Stream的衍生产品。

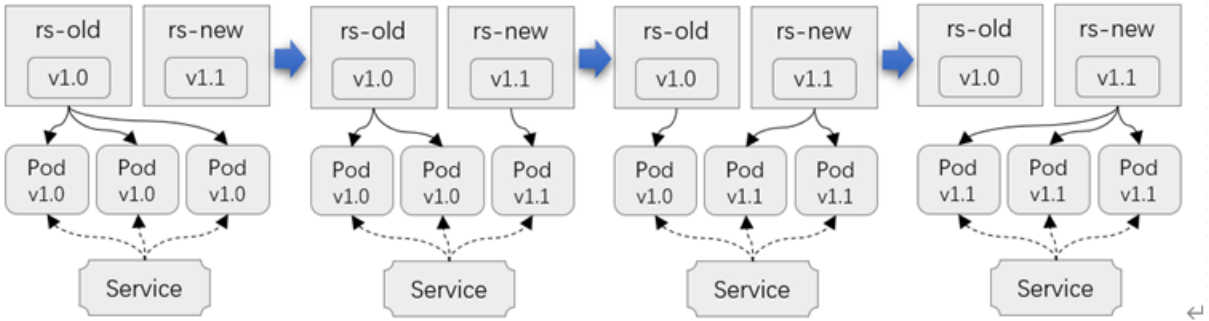
key: string, key: {}



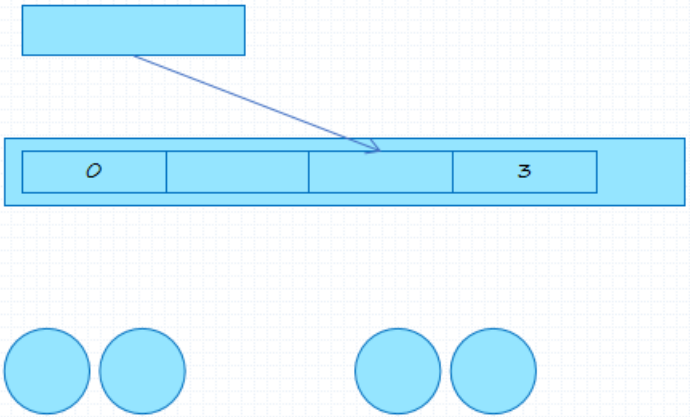
- env:
- - name:
- valueFrom:
- Secret



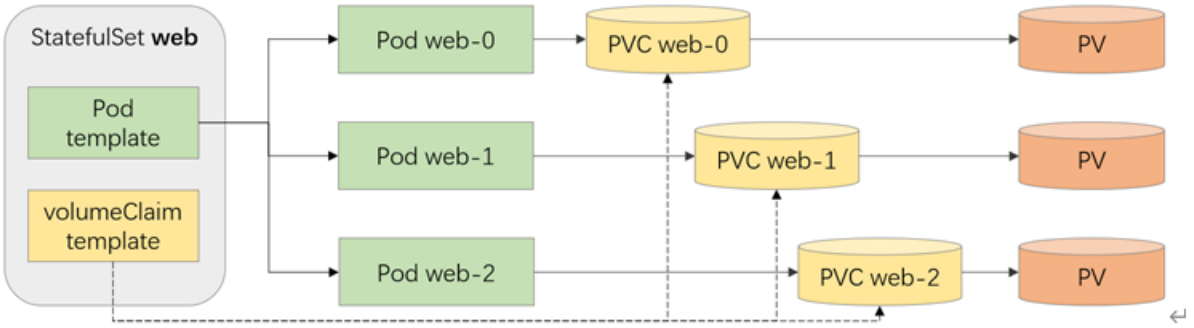


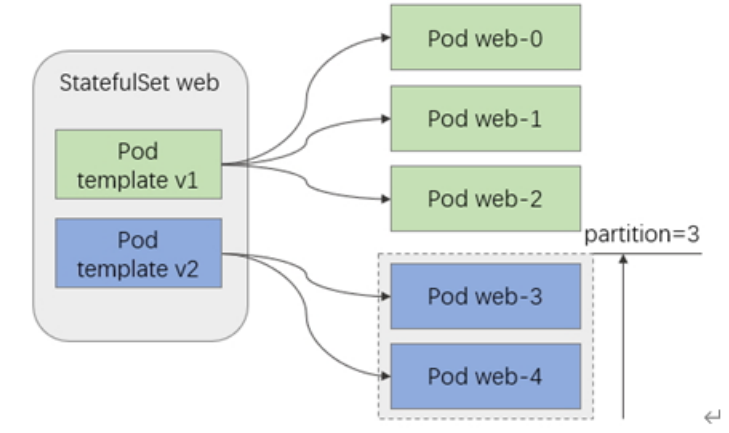


Deployment

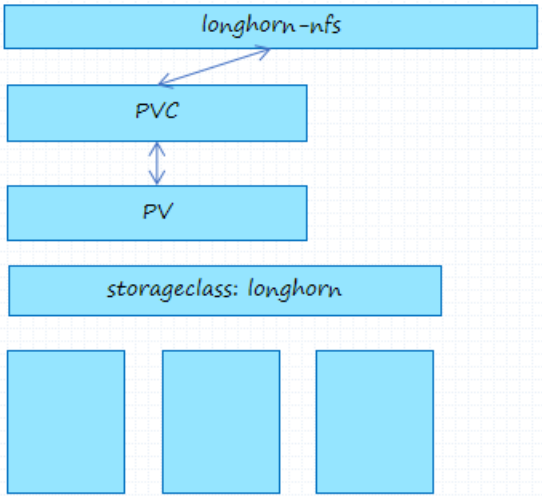


Job

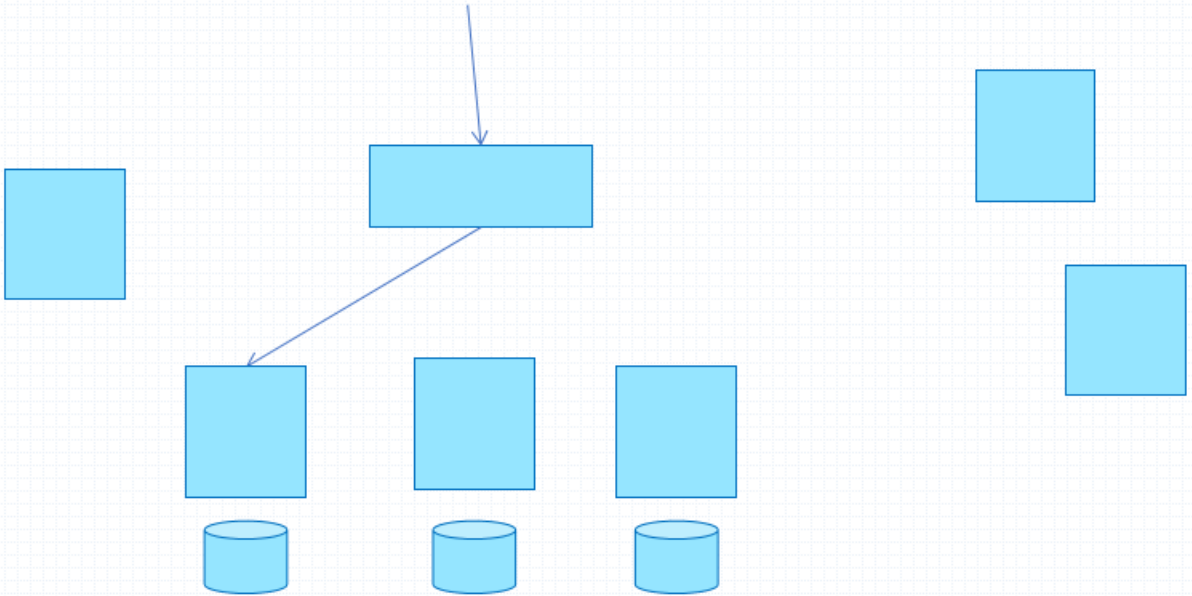



StatefulSet web
Pod template v1
Pod template v2
Pod web-0
Pod web-1
Pod web-2
partition=3
Pod web-3
Pod web-4

- /dir1, /dir2, ...



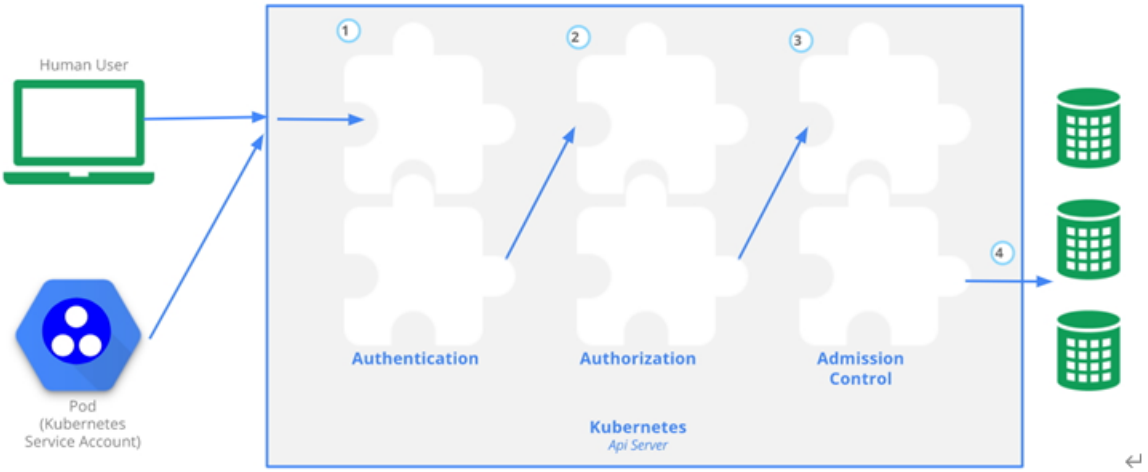
http协议无状态，但很多http应用有状态

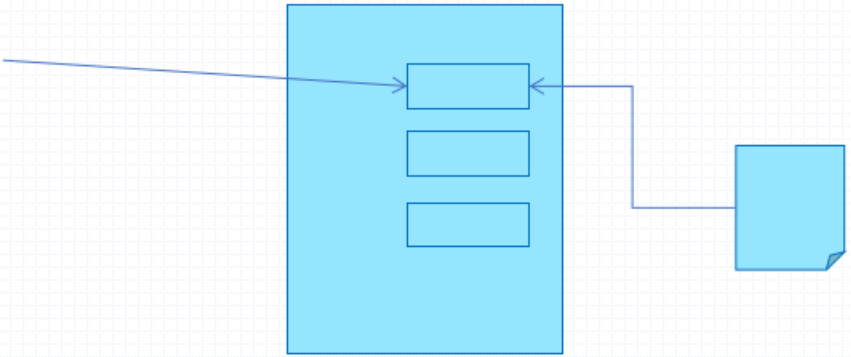


- StatefulSet: 通用的有状态应用控制器
  - 每个Pod都有自己的惟一标识，故障时，它只能被拥有同一标识的新实例所取代；
    - ${STATEFULSET_NAME}-${ORDINAL}, web-0, web-1, web-2,
    - Headless Service
  - 如果有必要，可以为被Pod配置专用的存储卷，且只能是PVC格式；

认证，授权和准入控制

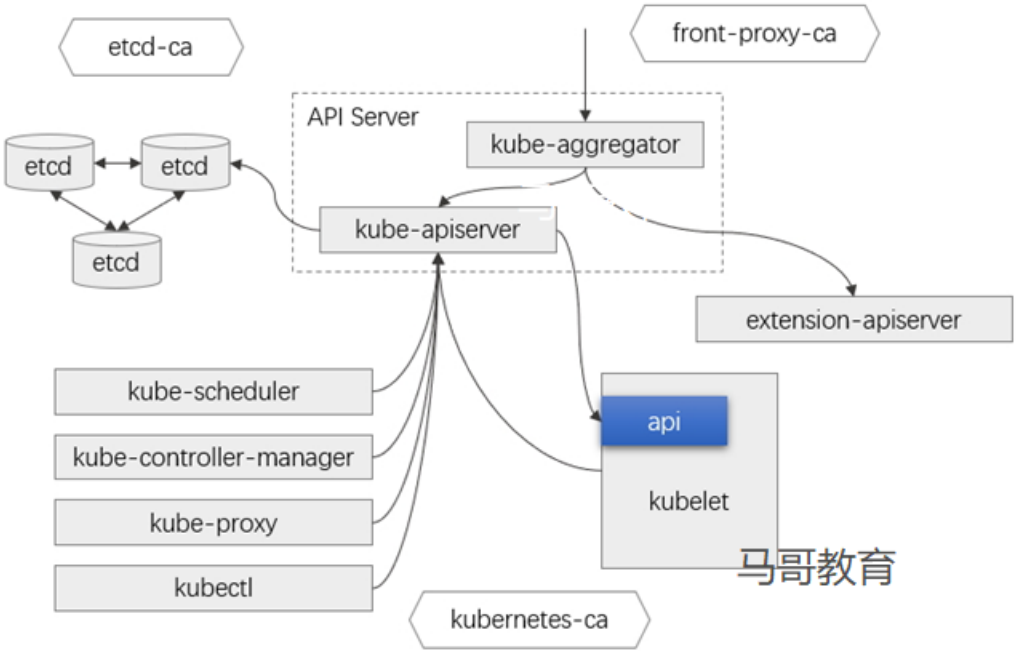
- Authn
- Authz
- Admission: 检验、变异，只发挥“写请求“上



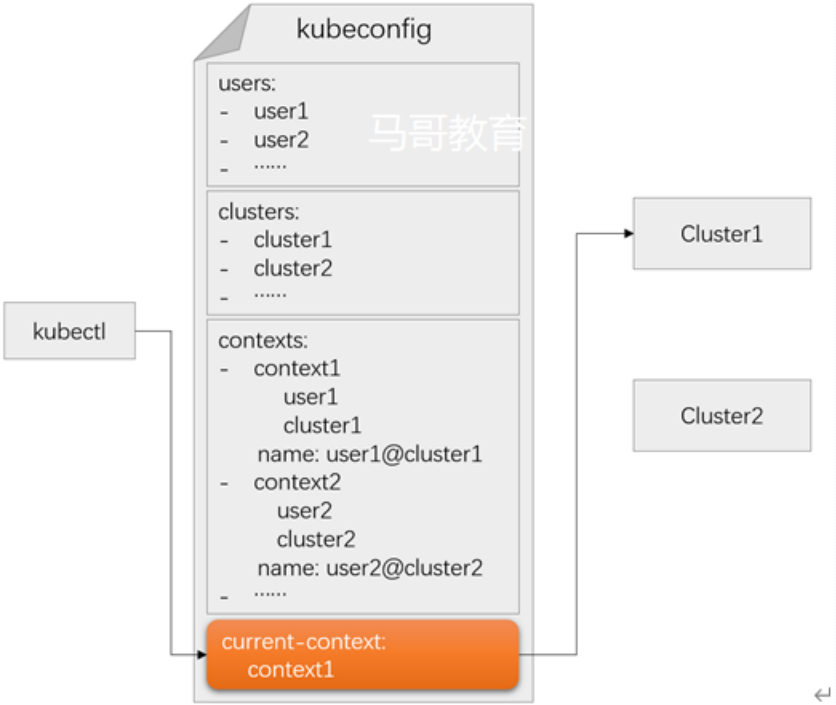


Vault

- https:
  - API Server
    - Client Cert:
      - ✓ CN: User Name
      - ✓ O: Group Name


etcd-ca
front-proxy-ca
API Server
kube-aggregator
etcd
etcd
etcd
kube-apiserver
extension-apiserver
kube-scheduler
kube-controller-manager
kube-proxy
kubectl
api
kubelet
马哥教育
kubernetes-ca

Default CN	Parent CA	O (in Subject)	kind	hosts (SAN)
kube-etcd	etcd-ca		server, client	`localhost`, `127.0.0.1`
kube-etcd-peer	etcd-ca		server, client	`<hostname>`, `<Host_IP>`, `localhost`, `127.0.0.1`
kube-etcd-healthcheck-client	etcd-ca		client	
kube-apiserver-etcd-client	etcd-ca	system:masters	client	
kube-apiserver	kubernetes-ca		server	`<hostname>`, `<Host_IP>`, `<advertise_IP>`
kube-apiserver-kubelet-client	kubernetes-ca	system:masters	client	
front-proxy-client	kubernetes-front-proxy-ca		client	



- v1.20
  - Node, RBAC

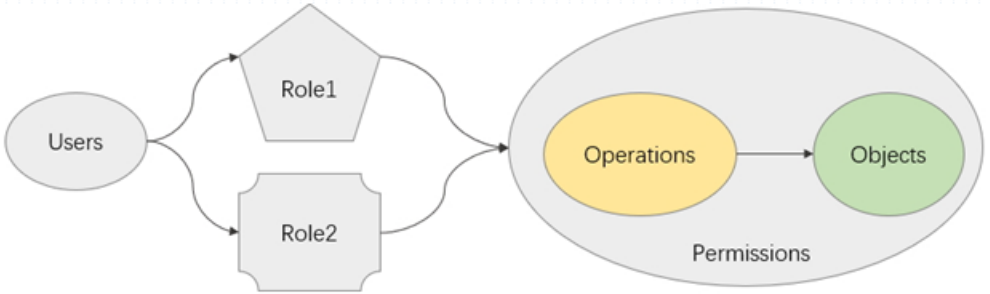


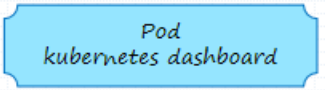
图 9-8 RBAC 中的用户、角色及权限

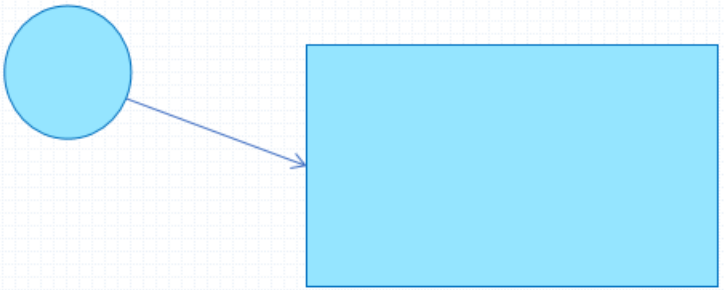
API Server 是 RESTful 风格的 API，各类客户端基于 HTTP 协议的请求报文（首部）发送身份认证信息并由认证插件完成身份验证，而后通过 HTTP 协议的请求方法指定对目标对象的操作请求并由授权插件进行授权检查，而操作的对象则是借助 URL 路径指定的 REST 资源。表 9-1 对比给出了 HTTP 方法和 Kubernetes API Server 资源操作的对应关系。↵

表 9-1 HTTP Method 与 API Server Verb↵

HTTP verb	request verb
POST	create
GET, HEAD	get (for individual resources), list (for collections)
PUT	update
PATCH	patch
DELETE	delete (for individual resources), deletecollection (for collections)

- RBAC
  - Role, ClusterRole
  - RoleBinding, ClusterRoleBinding
- Verbs, Object
- Subject





利用 Role 和 ClusterRole 两类角色进行赋权时需要用到另外两种资源 RoleBinding 和 ClusterRoleBinding，它们同样是由 API Server 内置支持资源类型。RoleBinding 用于将 Role 绑定到一个或一组用户之上，它隶属于且仅能作用于其所在的单个名称空间。RoleBinding 可以引用同一名称中的 Role，也可以引用集群级别的 ClusterRole，但引用的 ClusterRole 的许可能权限会降级到仅生效于 RoleBinding 所在的名称空间。而 ClusterRoleBinding 则是用于将 ClusterRole 绑定到用户或组，它作用于集群全局，且仅能够引用 ClusterRole。如图 9-10 所示。

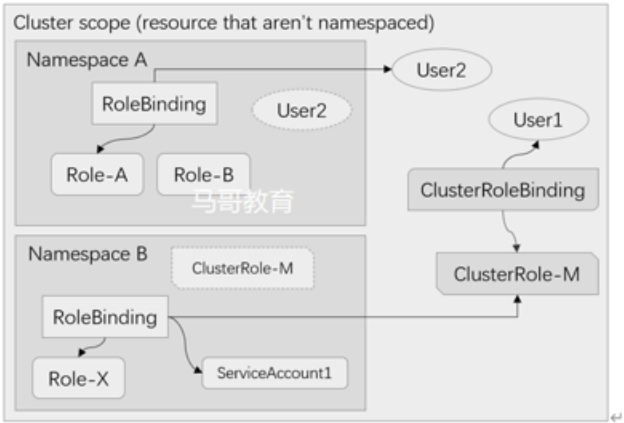


图 9-10 Role、RoleBinding、ClusterRole 和 ClusterRoleBinding

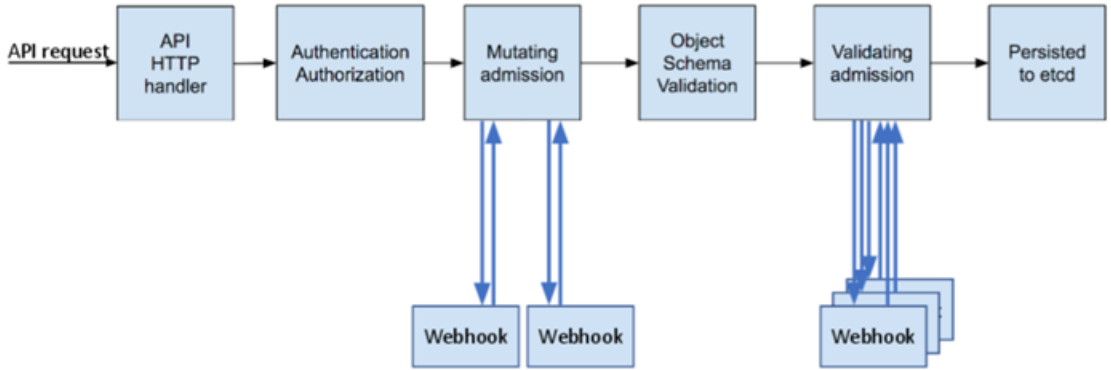


图 9-15 准入控制器的运行阶段 （图片来源：https://kubernetes.io/）

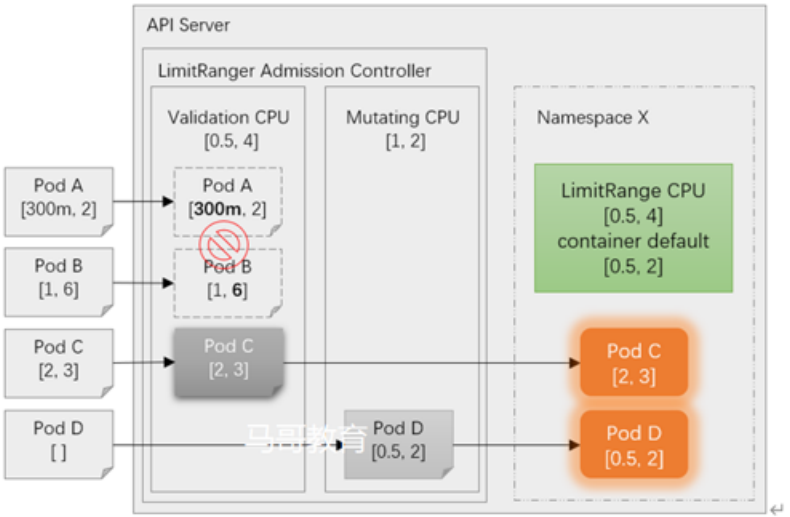


图 9-16 LimitRange 和 LimitRanger 示意图



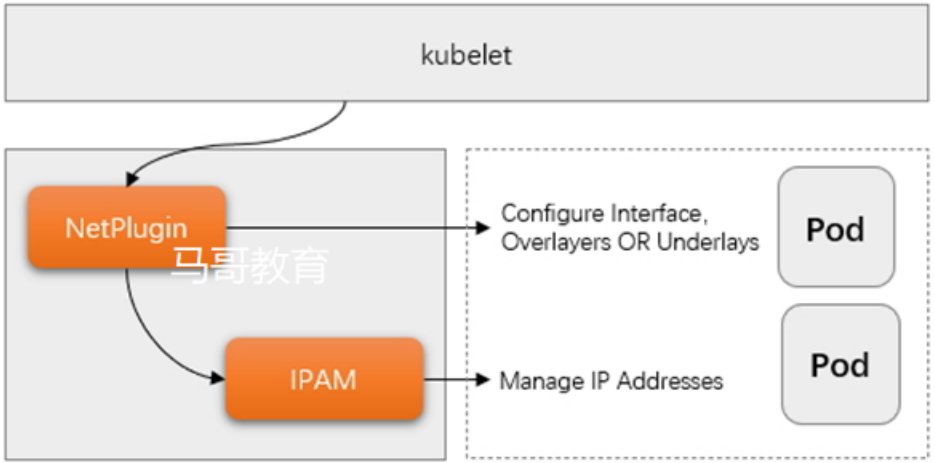



图 10-5　CNI 插件 API

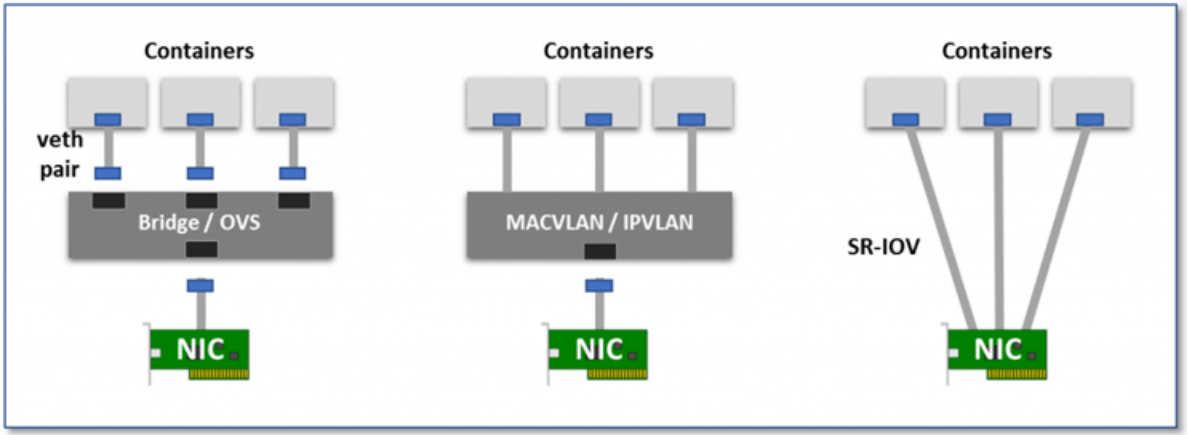


图 10-6 虚拟网桥、多路复用及硬件交换

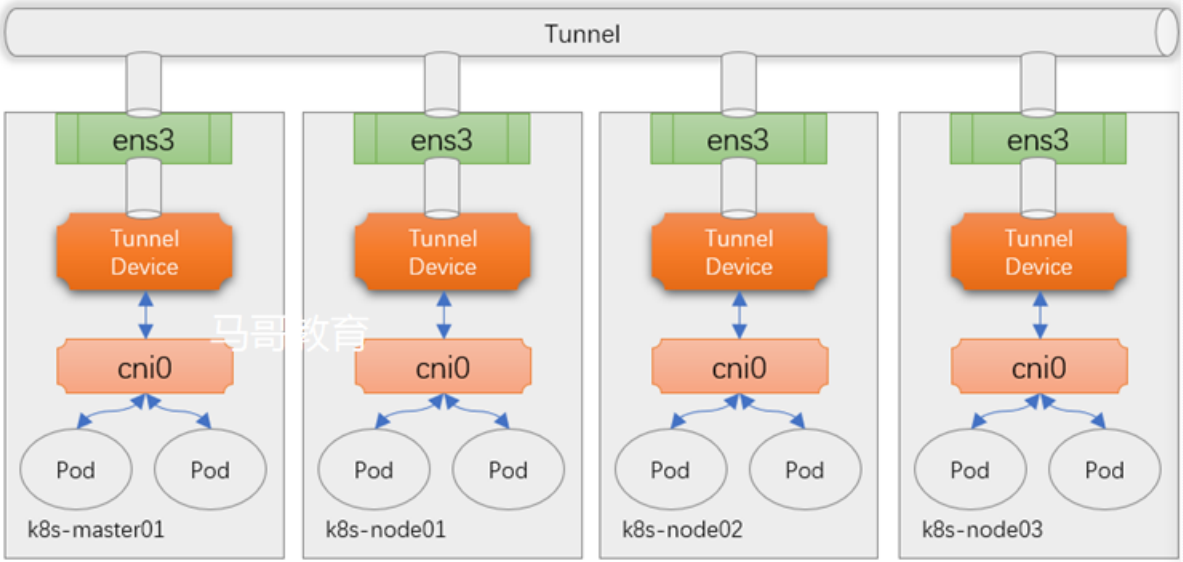
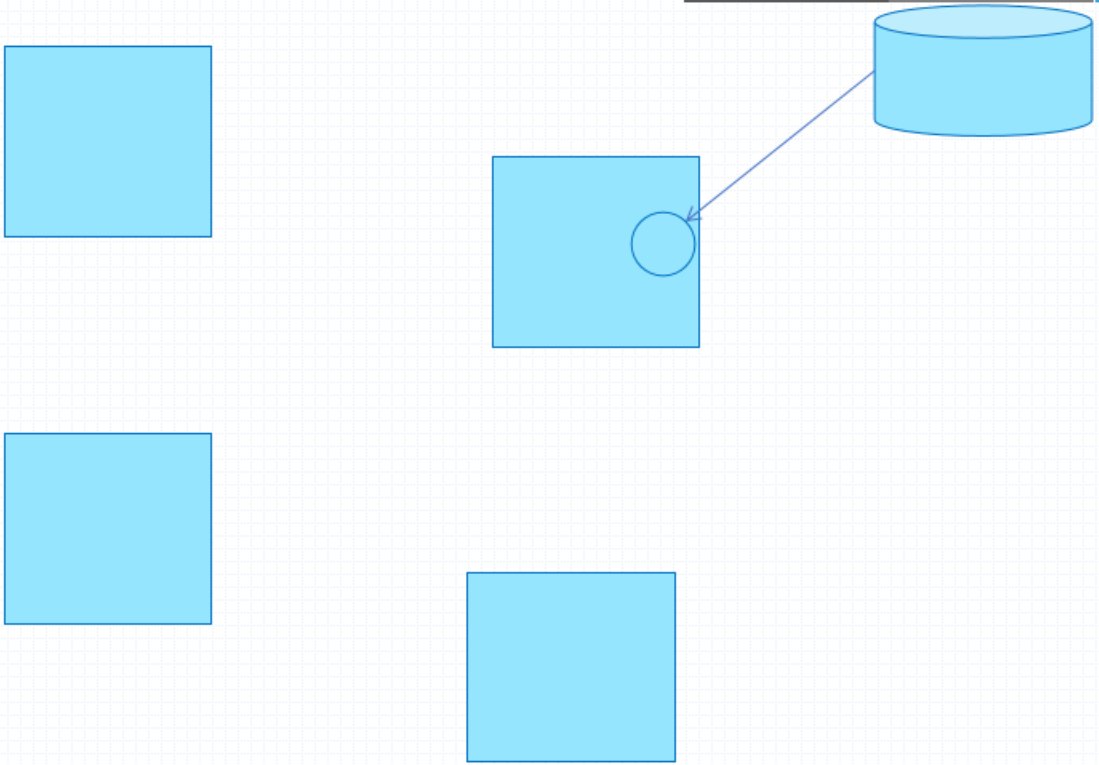
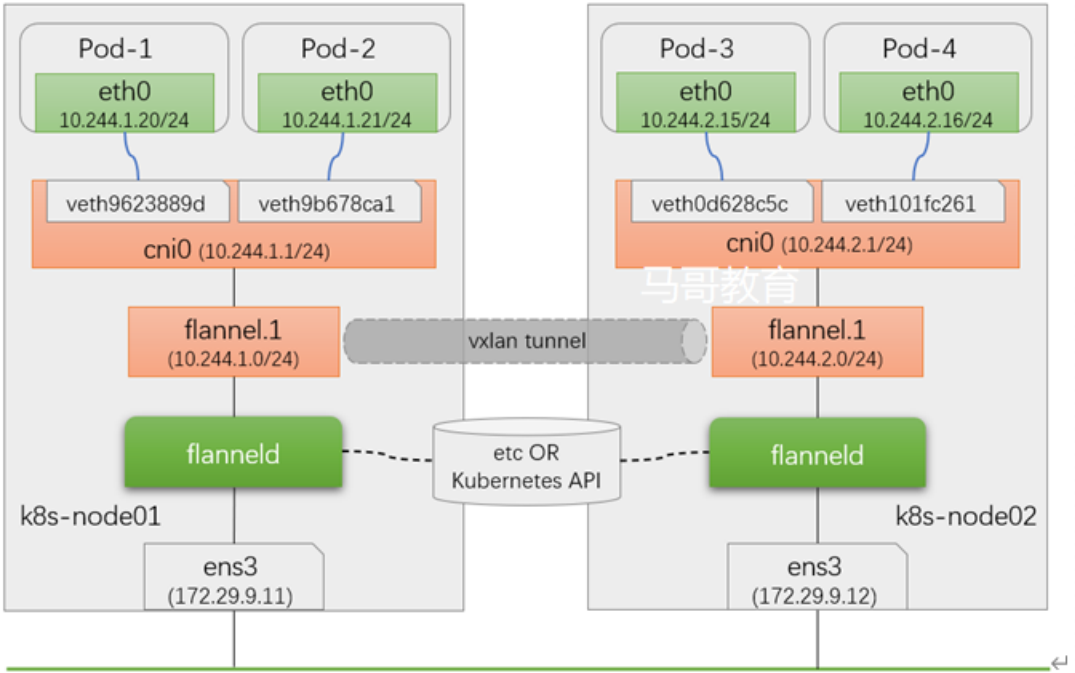


图 10-4 叠加网络功能示意图↵



flannel VXLAN



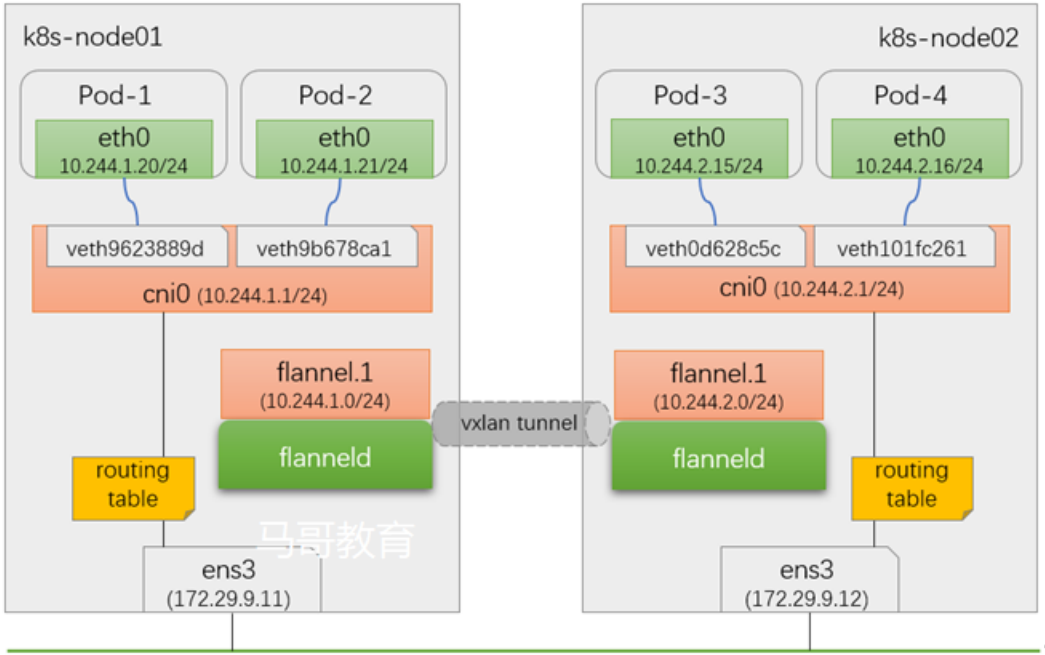


图 10-13 VXLAN DirectRouting 模式中的 Pod 间通信

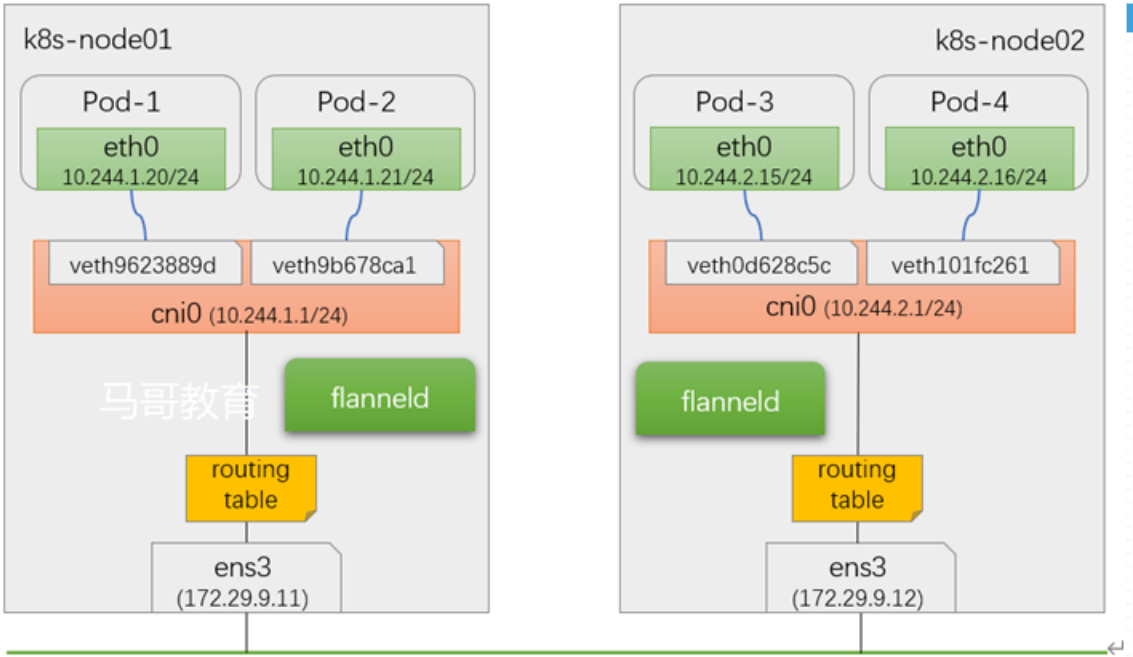


图 10-14 host-gw 后端

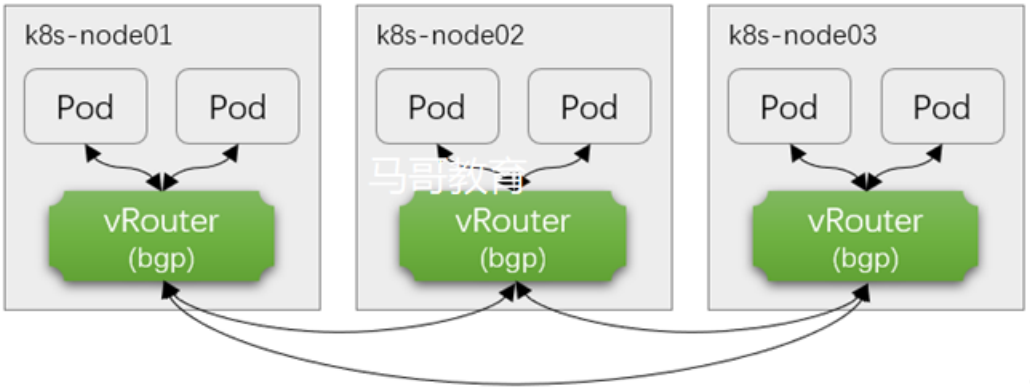


图 10-15 Calico 系统示意图

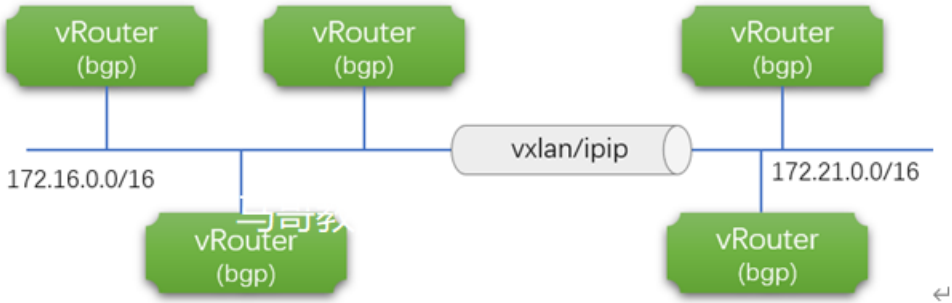
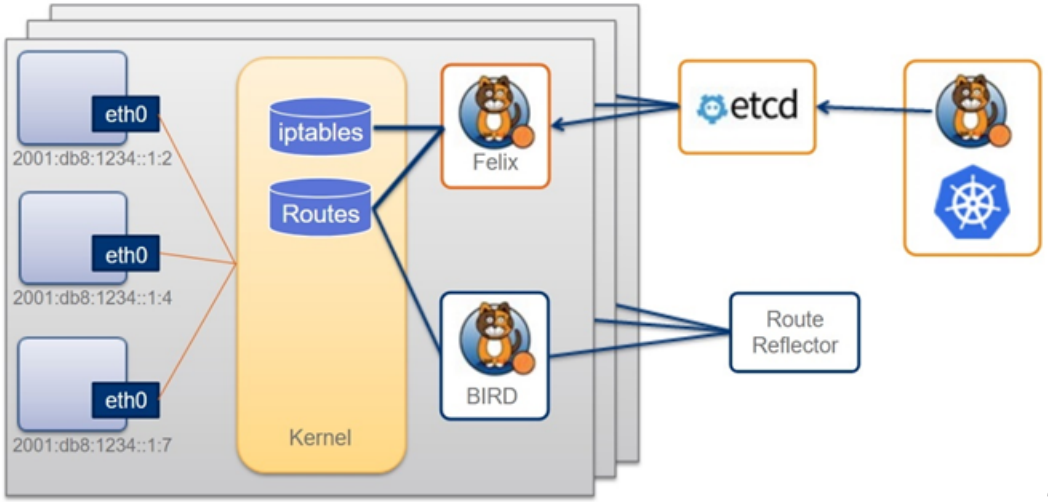


图 10-16 Calico 混合通信模型


eth0
2001:db8:1234::1:2
eth0
2001:db8:1234::1:4
eth0
2001:db8:1234::1:7
iptables
Routes
Kernel
Felix
BIRD
etcd
Route
Reflector

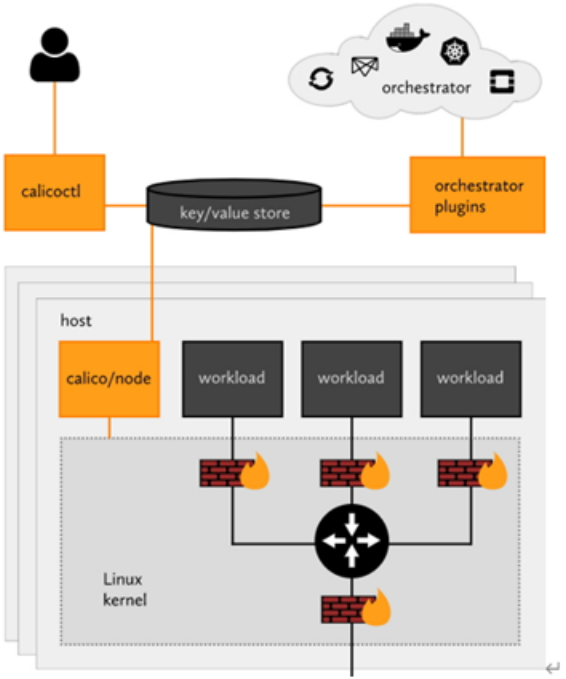


图 10-18 Calico 程序组件

- calico/node：Calico于Kubernetes集群中每个节点运行的节点代理，负责提供felix、bird4、bird6和confd等守护进程；
- calico/kube-controllers：Calico运行于Kubernetes之上的自定义控制器，它也是Calico协同Kubernetes的插件；

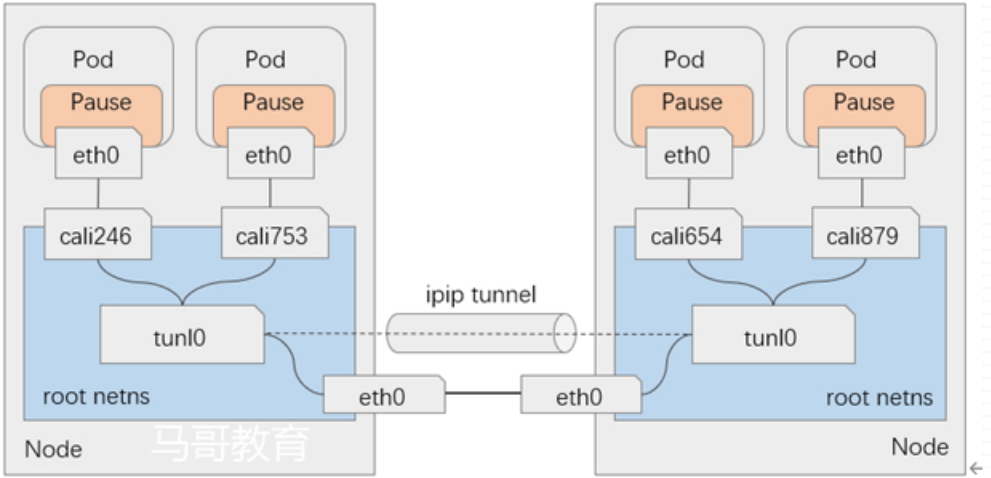
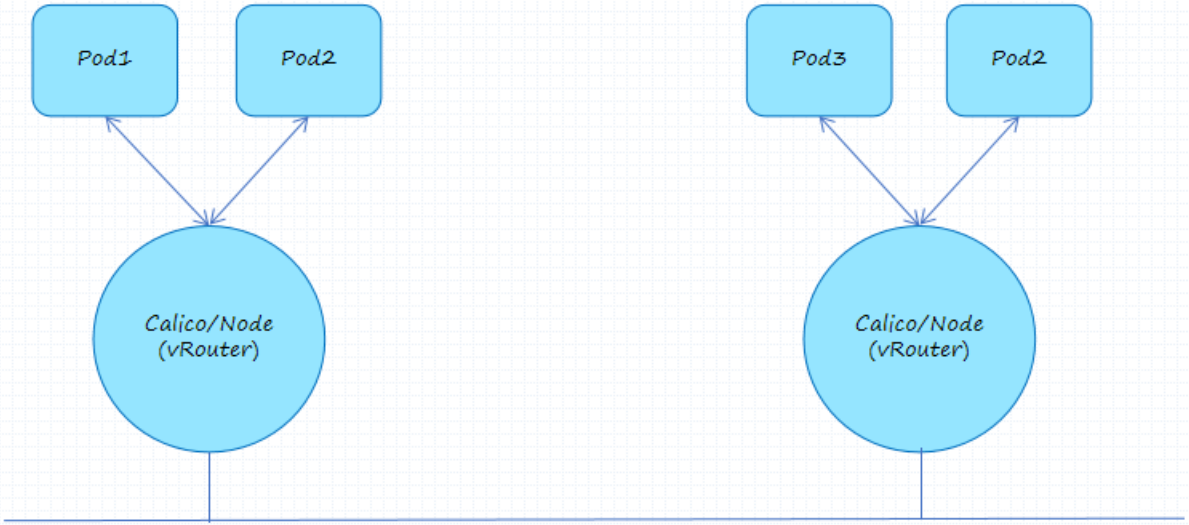


图 10-19 Calico IPIP

Flannel host-gw, Calico BGP

Pod1 IP, Pod3 IP



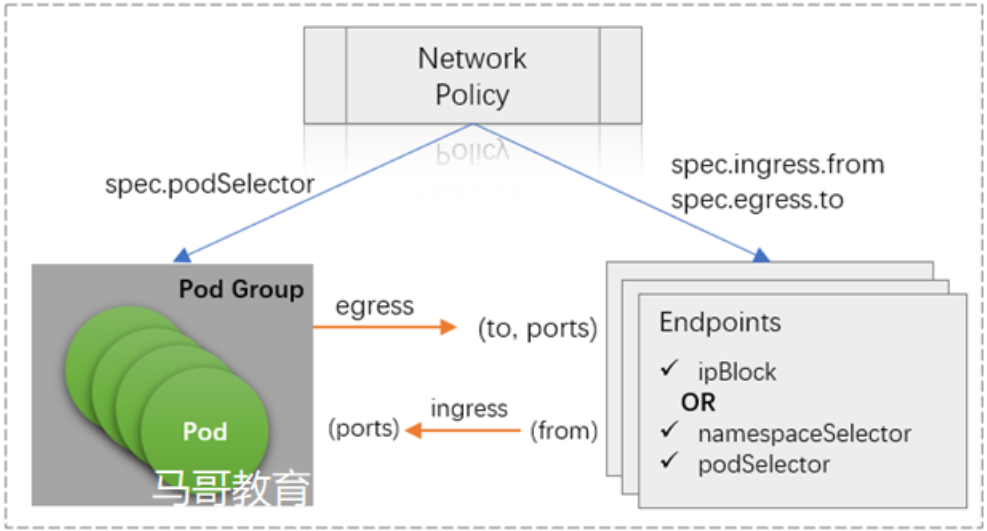
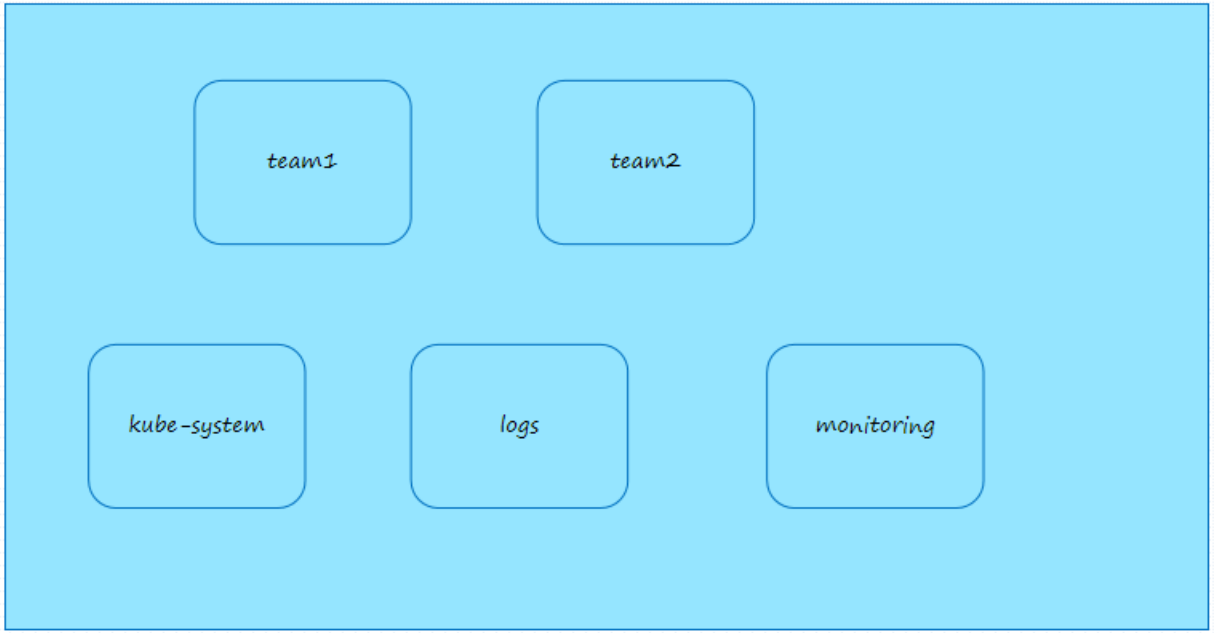


图 10-21 网络策略示意图


team1
team2
kube-system
logs
monitoring

default-scheduler

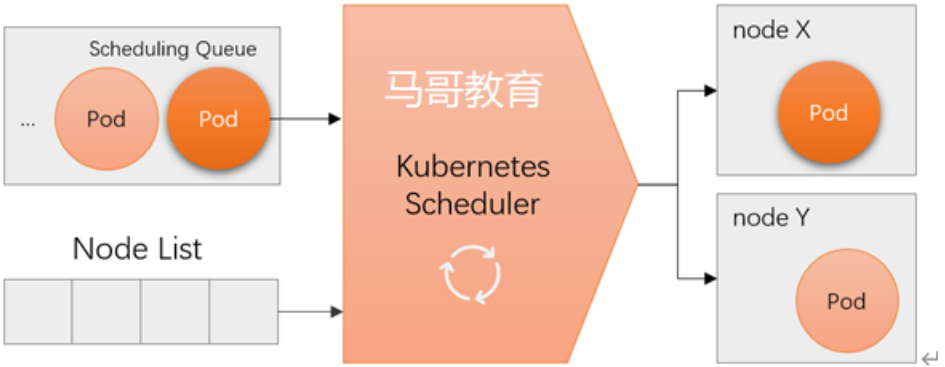


图 11-1 Kubernetes 调度操作示意图

- API Server：
- Controller Manager
- Scheduler：调度器
  - Pod：未绑定到任何节点，即为该Pod从众多节点中选出一个最佳节点；

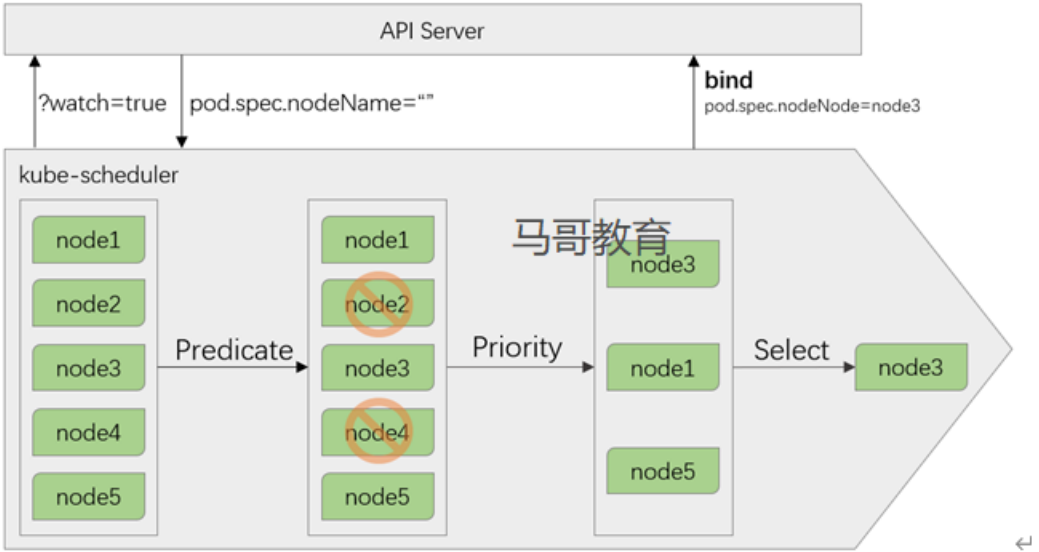


图 11-2 预选、优选及选定示意图

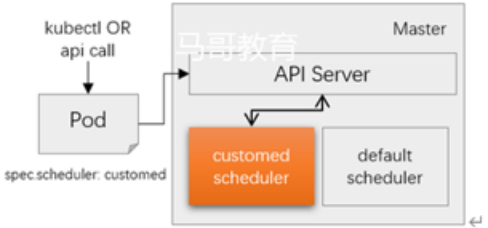


图 11-3 多调度器

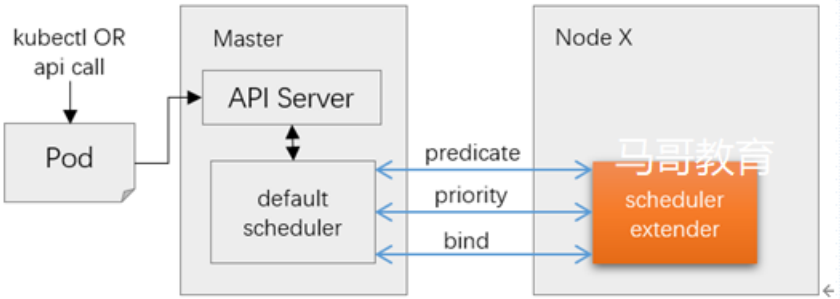


图 11-4 Scheduler Extender

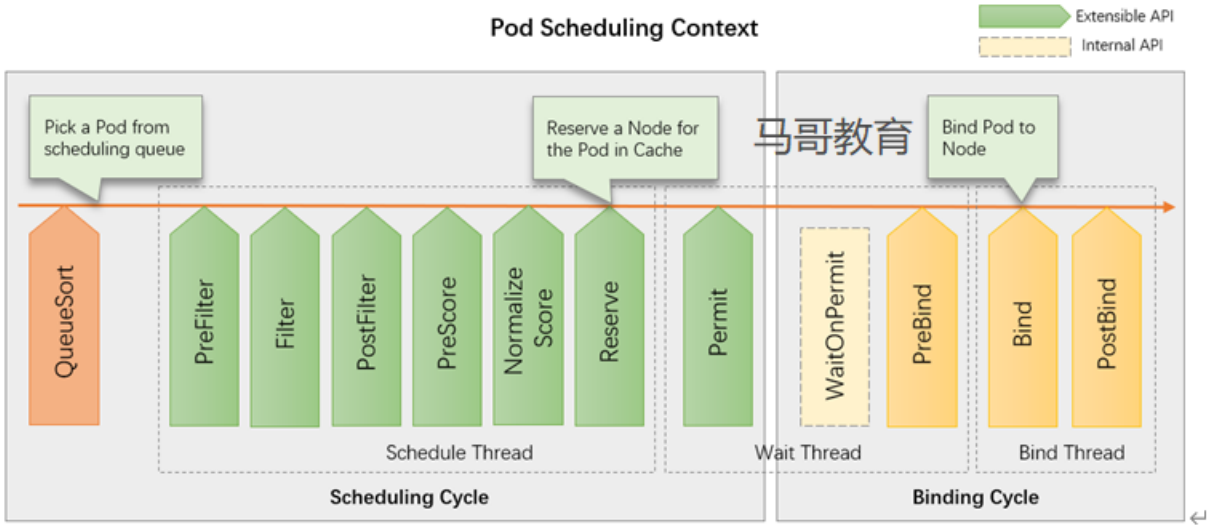
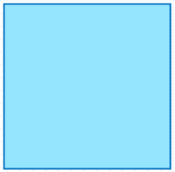
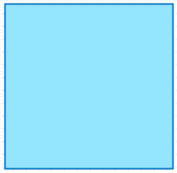
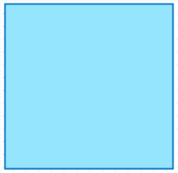
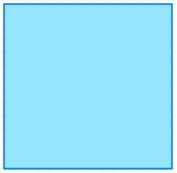


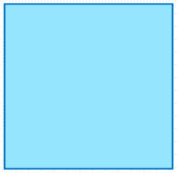
图 11-5 调度器扩展框架











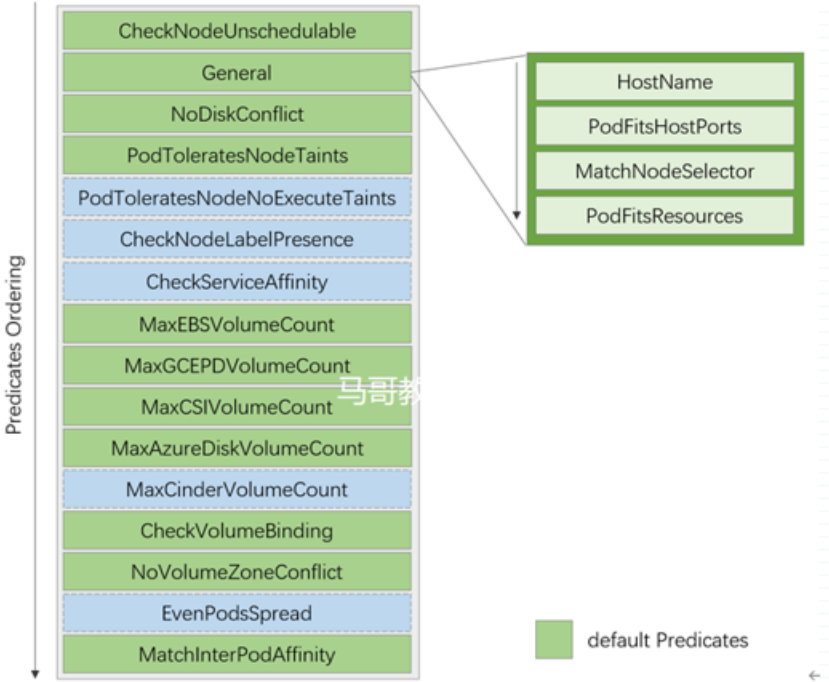


图 11-6 Kubernetes 的经典预选函数和应用次序

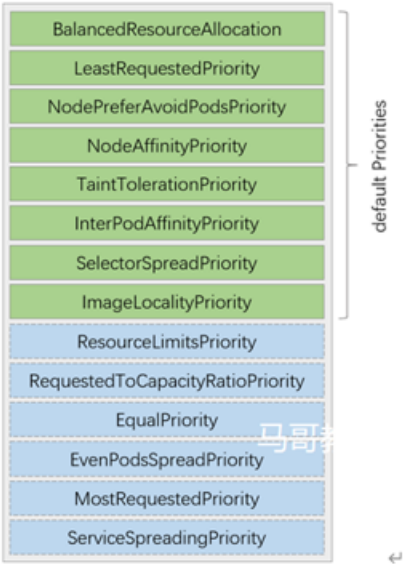


图 11-7 Kubernetes 的经典优选函数

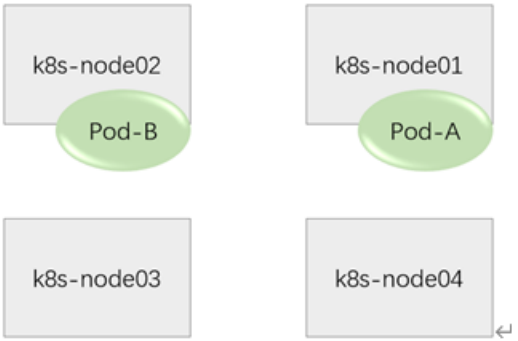


图 11-10 Pod 资源与位置拓扑

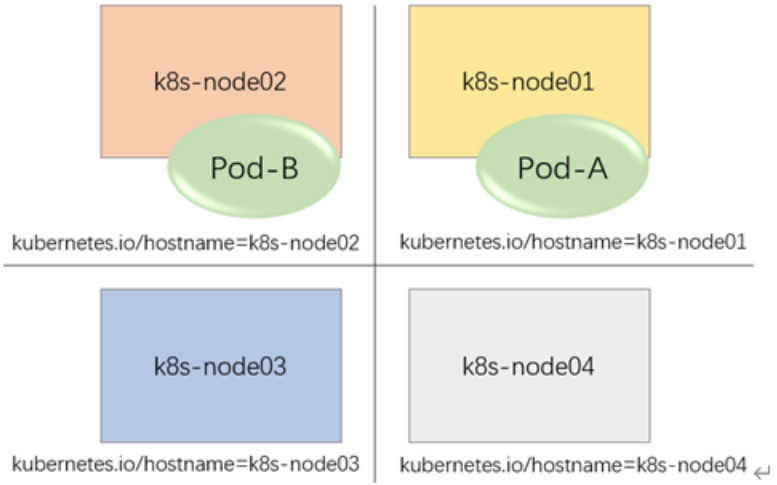


图 11-11 基于节点的位置拓扑

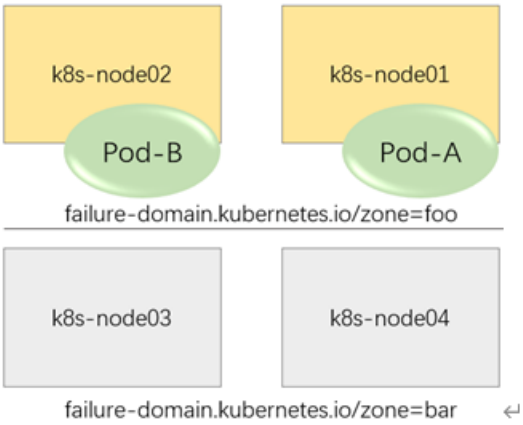
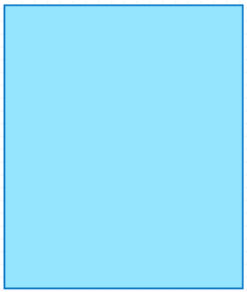
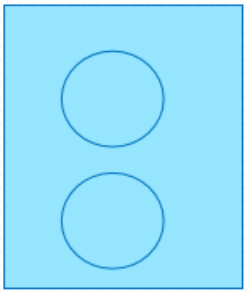


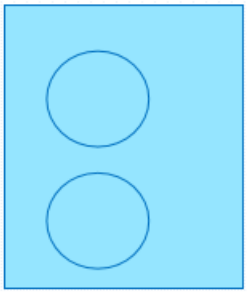
图 11-12 基于故障转域的位置拓扑

2-1=1

- 驱离， effect
- 极差







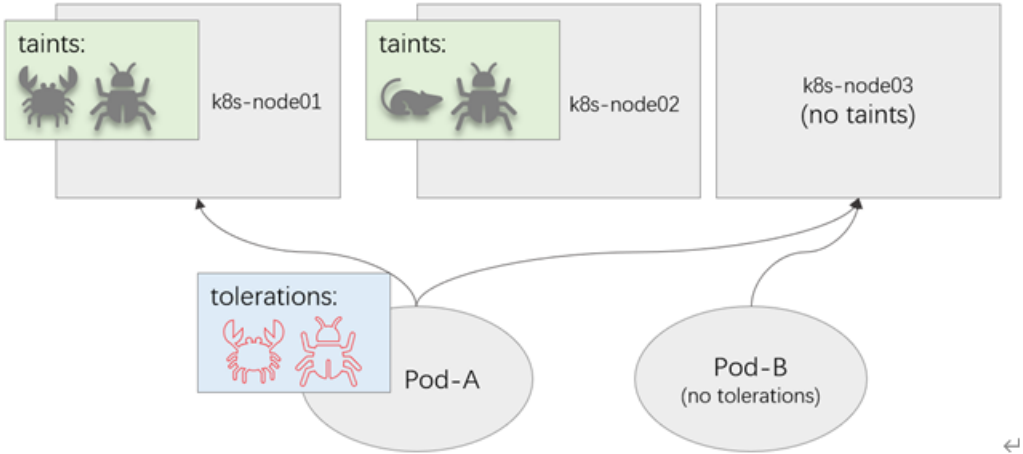


图 11-13 污点与容忍之间的关系示意图

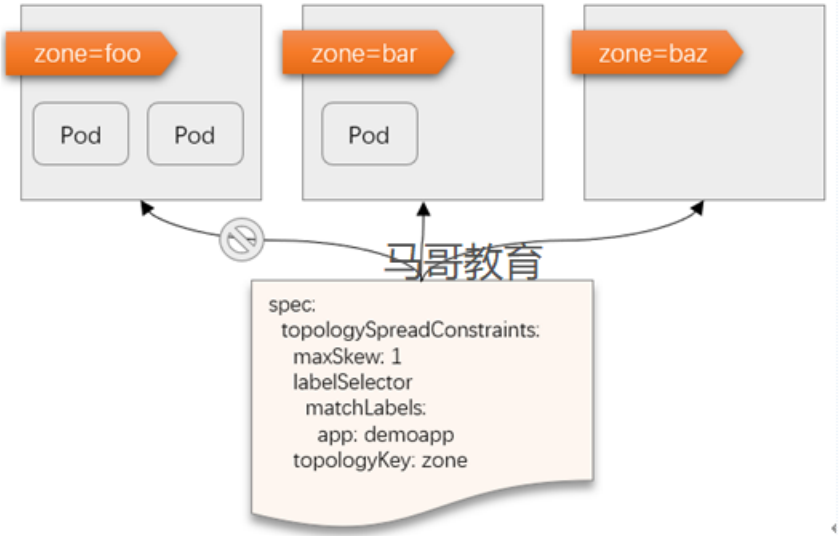


图 11-14 Pod 拓扑分布式调度示例

PaaS

- longhorn, calico
  - CRD: CustomResourceDefinition
  - API
  - 业务代码， 分布式
    - 基础设施服务: 消息队列， DB，

Resource Kind --> Controller(代码)

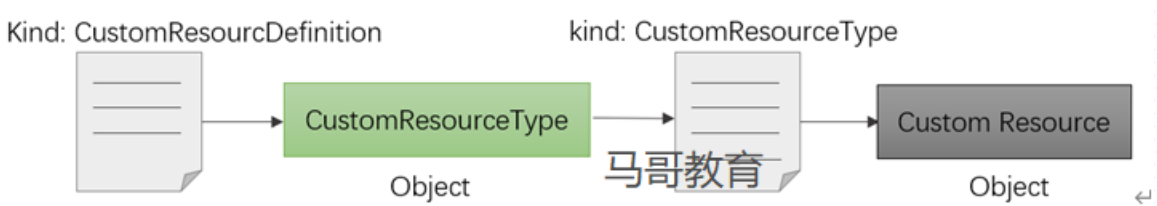


图 12-1 创建自定义资源类型及自定义类型对象

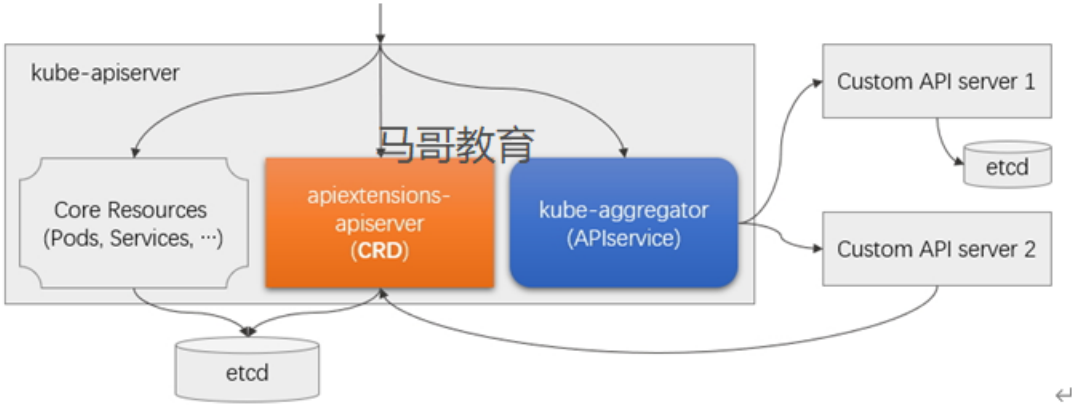


图 12-2 自定义 API 服务器及 APIService 资源

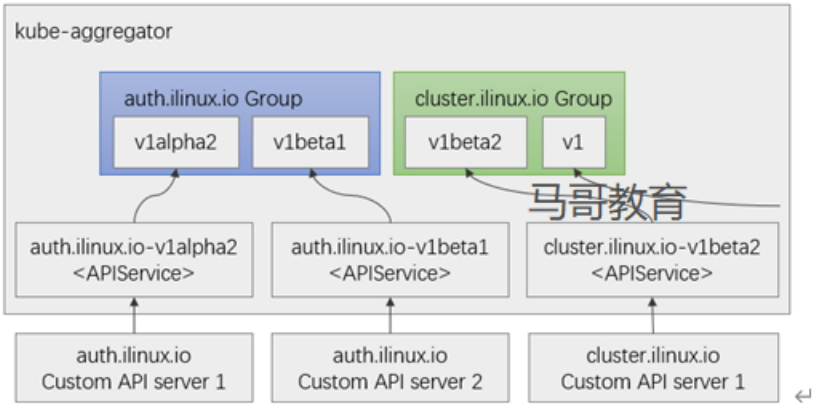


图 12-3　Kubernetes 聚合层及其聚合方式

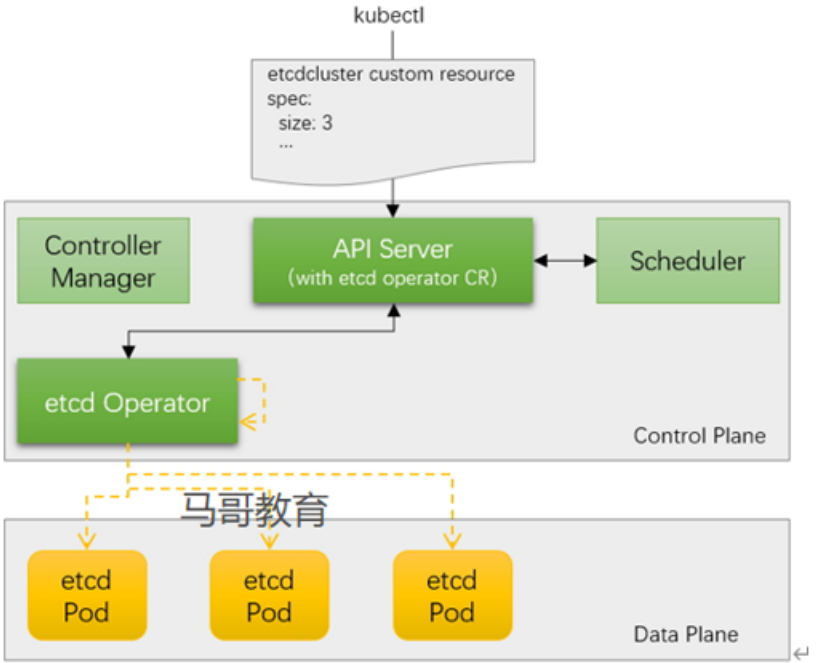


图 12-6 Operator 示意图

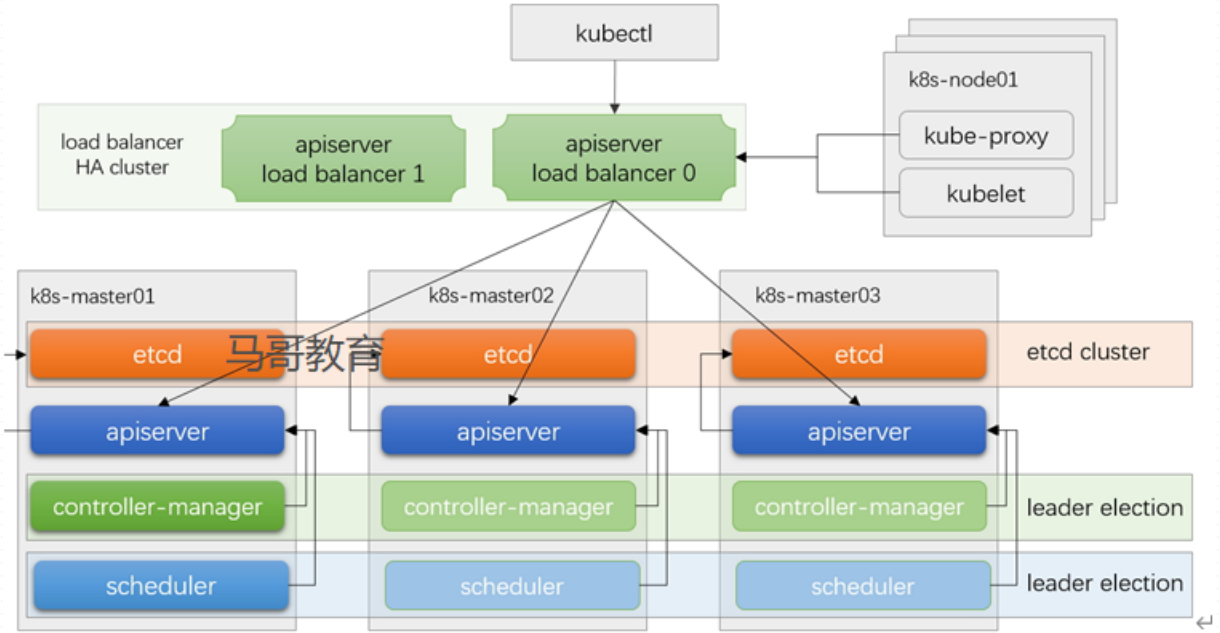
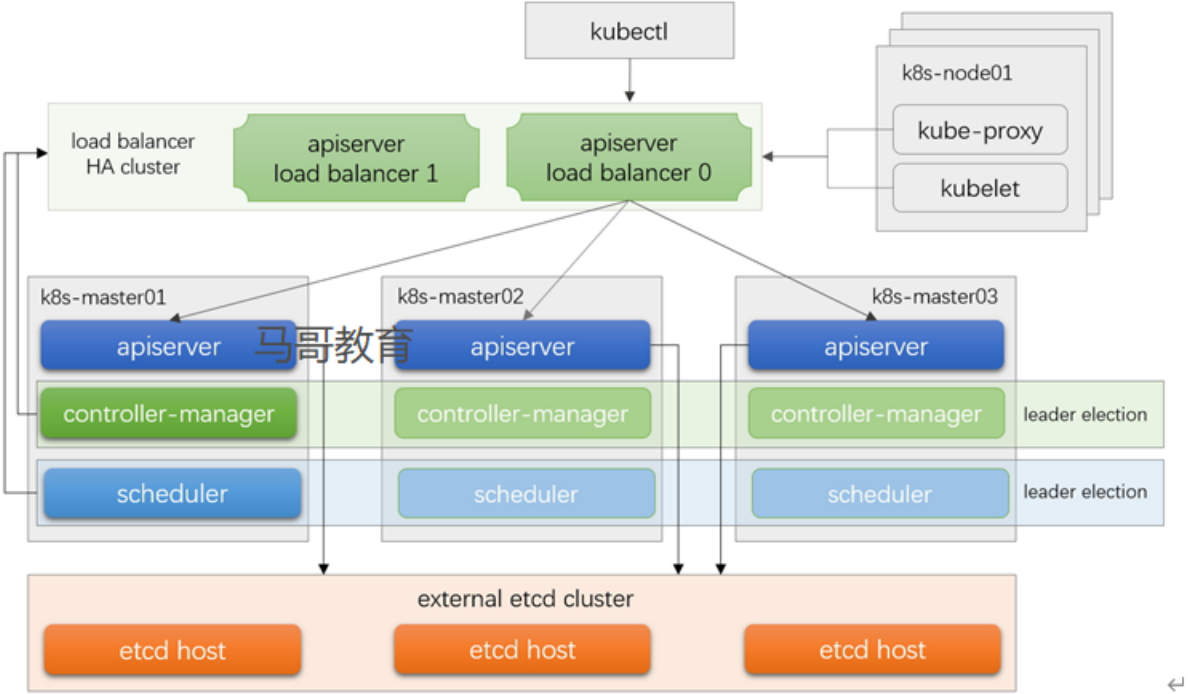
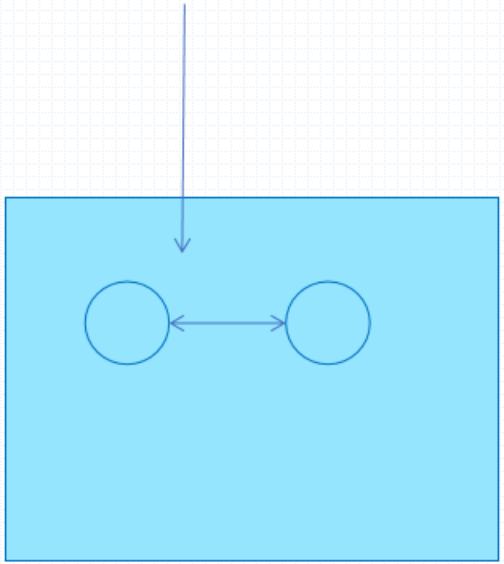


图 12-8 堆叠式 etcd



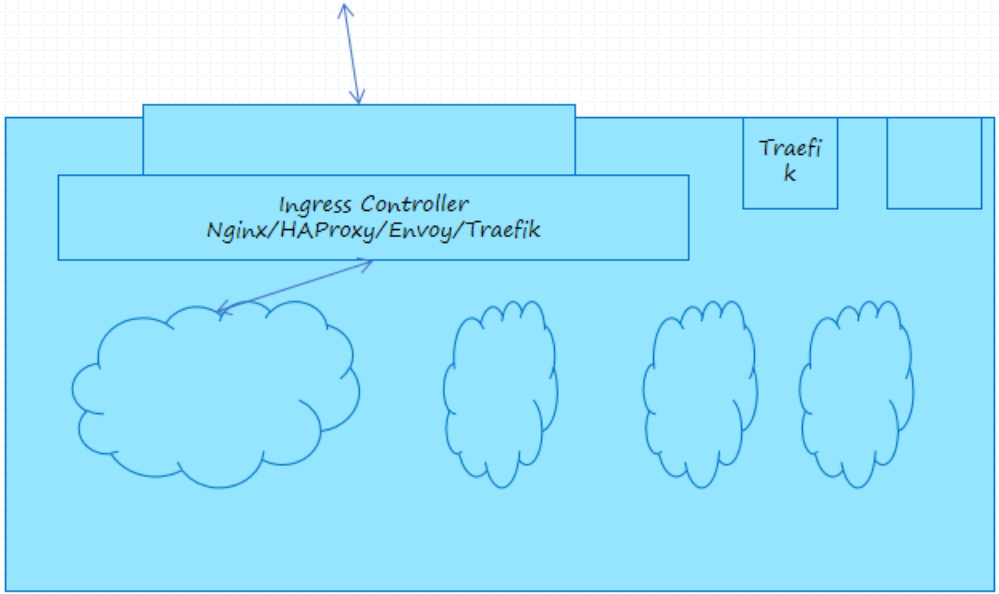
- Service
  - Service NodePort
  - Service externalIP
- Host
  - hostPort
  - hostNetwork



- Ingress: 集群外部注入集群内部的流量
- Egress： 集群内部流出到集群外部的流量

longhorn UI, Dashboard, Ingress

gw.magedu.com
/dashboard
/longhorn



Ingress-Nginx
HAProxy
Envoy
Traefik

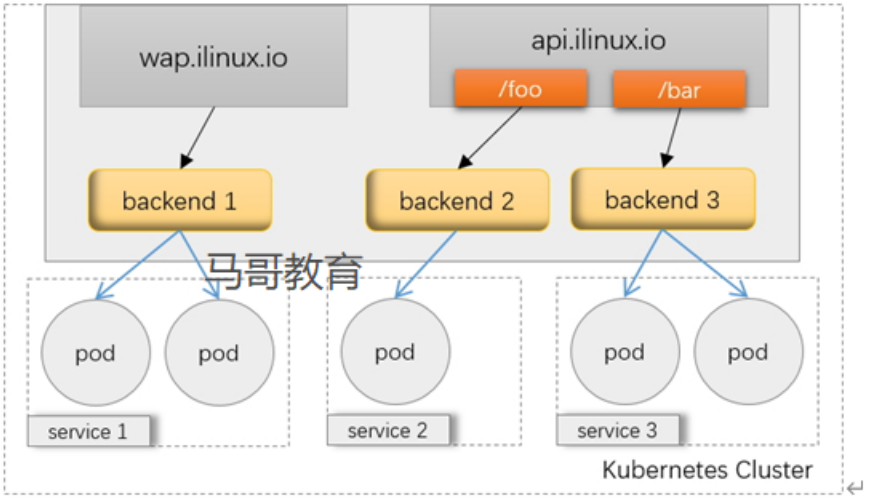
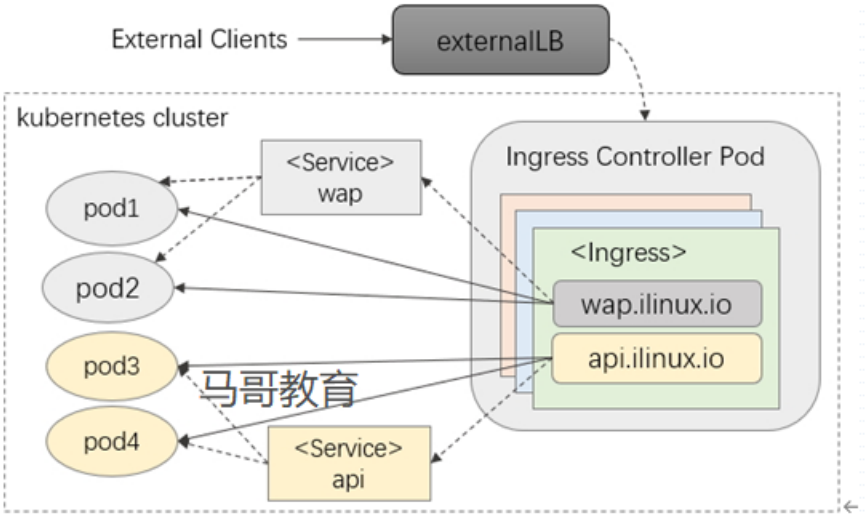


图 13-1 Ingress 资源示意图


External Clients
externalLB
kubernetes cluster
<Service>
wap
Ingress Controller Pod
pod1
pod2
<Ingress>
wap.ilinux.io
api.ilinux.io
pod3
马哥教育
pod4
<Service>
api

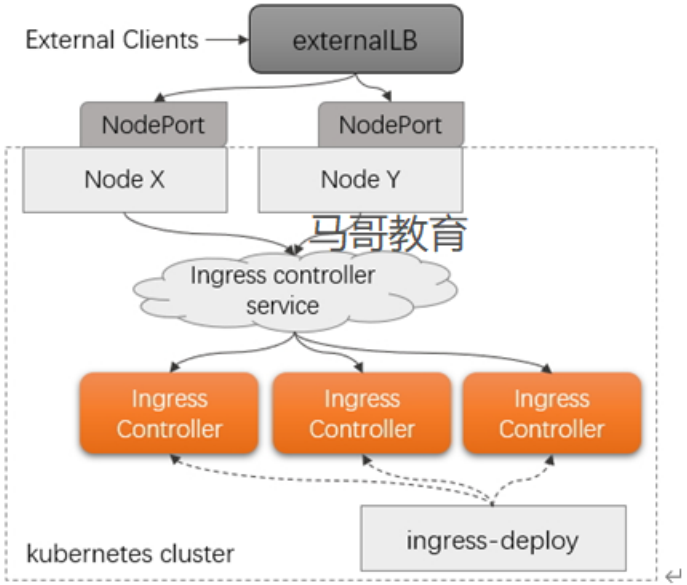
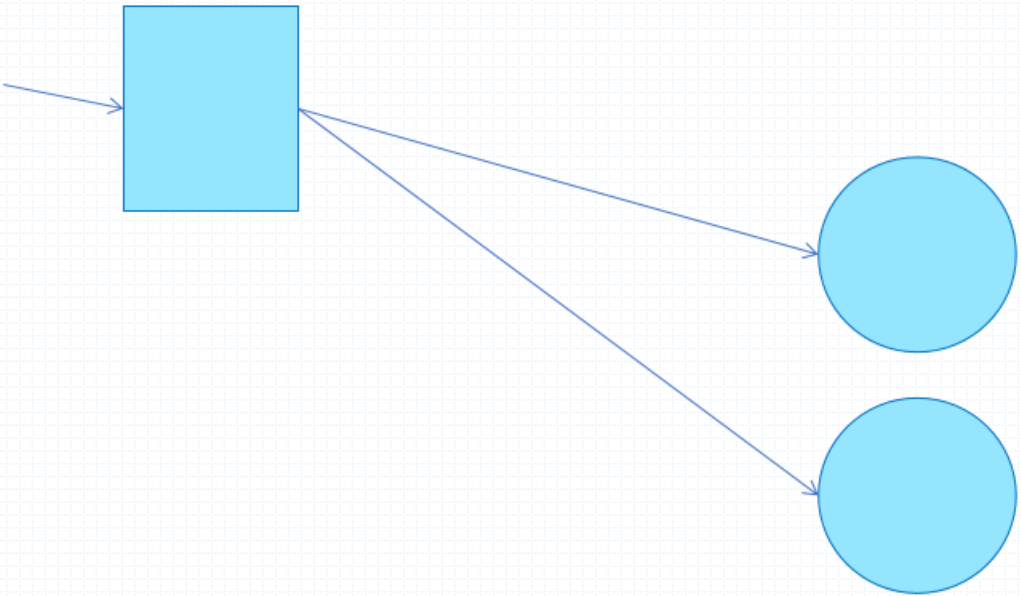


图 13-3 使用专用的 Service 对象为 Ingress 控制器接入外部流量

Gloo, Contour

- Management Server



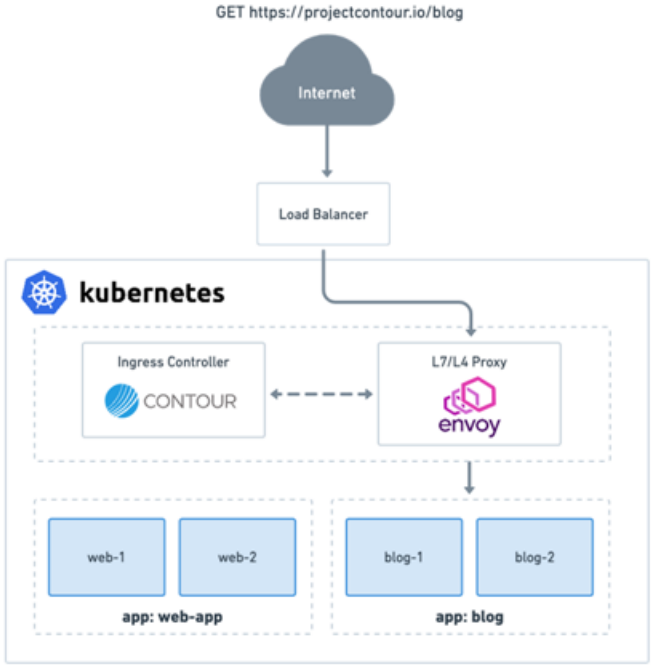


图 13-7 Contour 架构示意图

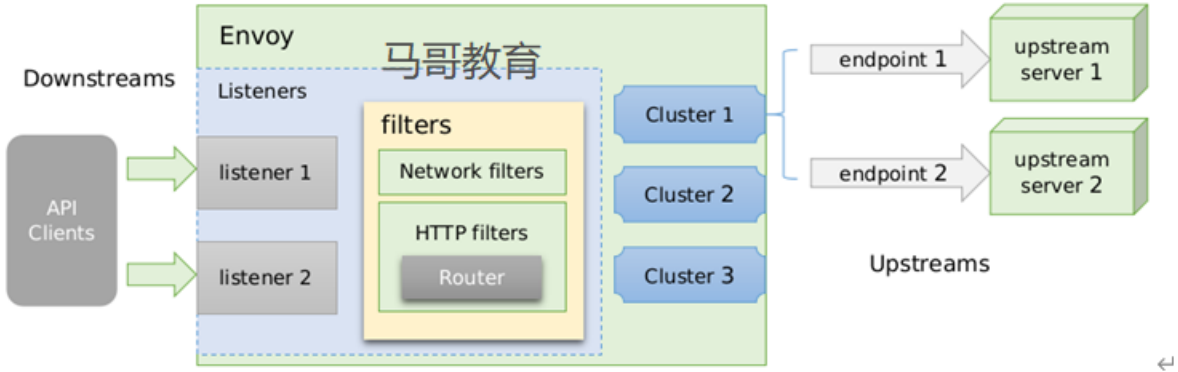


图 13-8 Envoy 代理中的核心逻辑组件

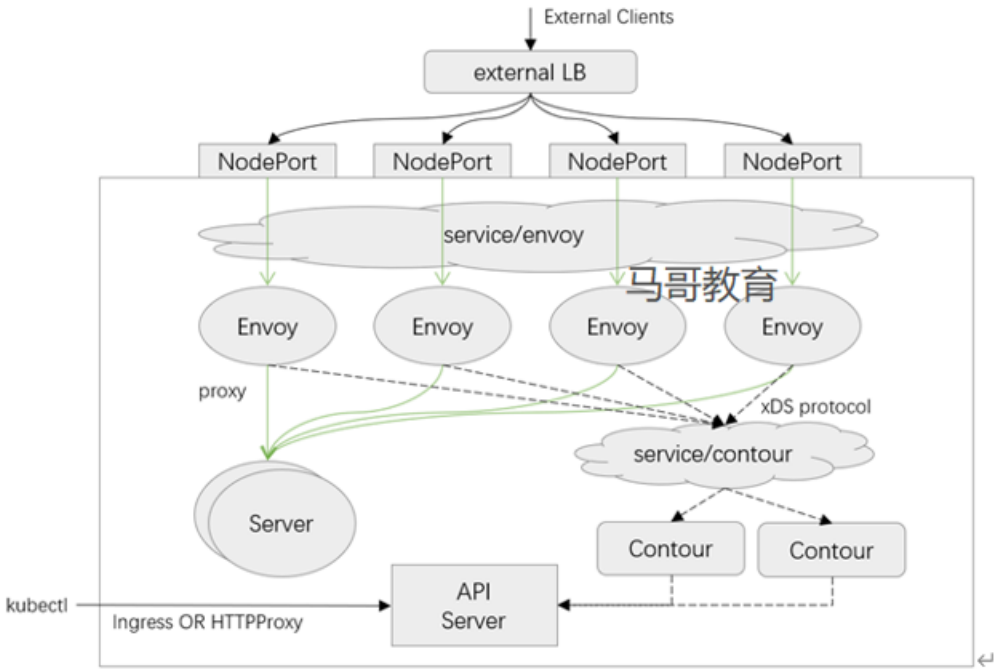


图 13-11 Contour 默认部署的各组件及工作逻辑示意图

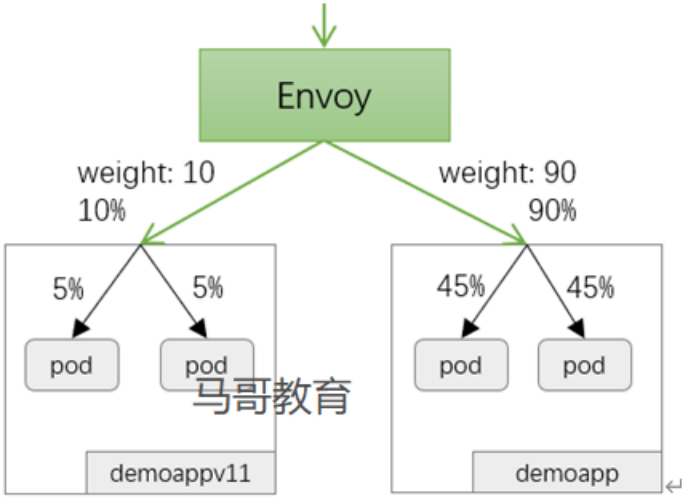
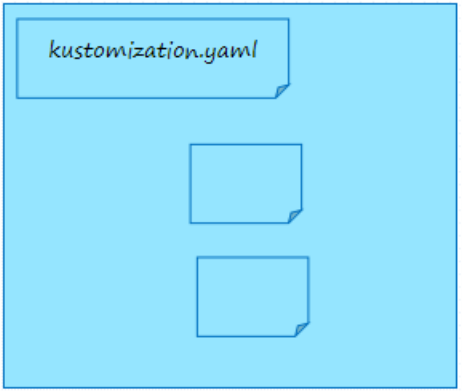


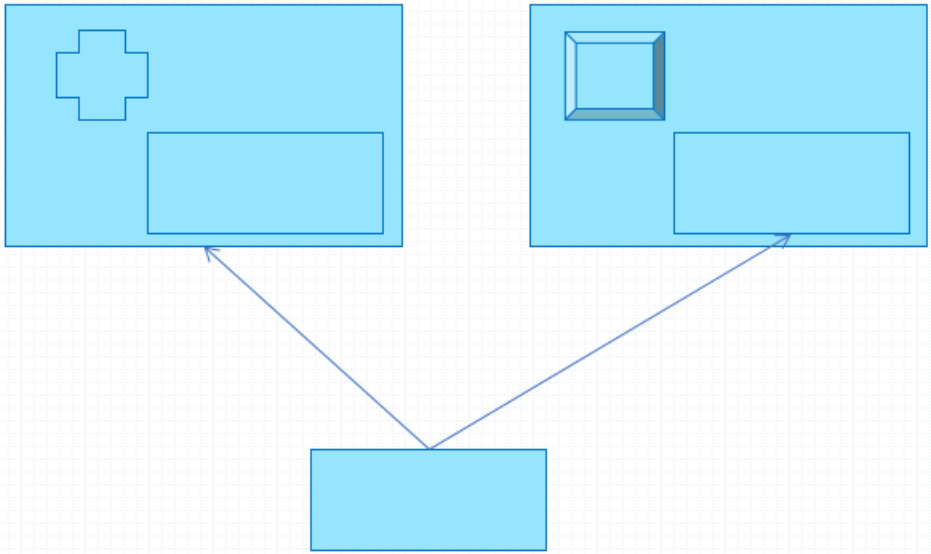
图 13-12 流量分割示意图

- Prometheus Server +
  - TSDB
- StatefulSet, PVC
- HA, k8s
- alertmanager, statefulset, ...

应用程序部署管理

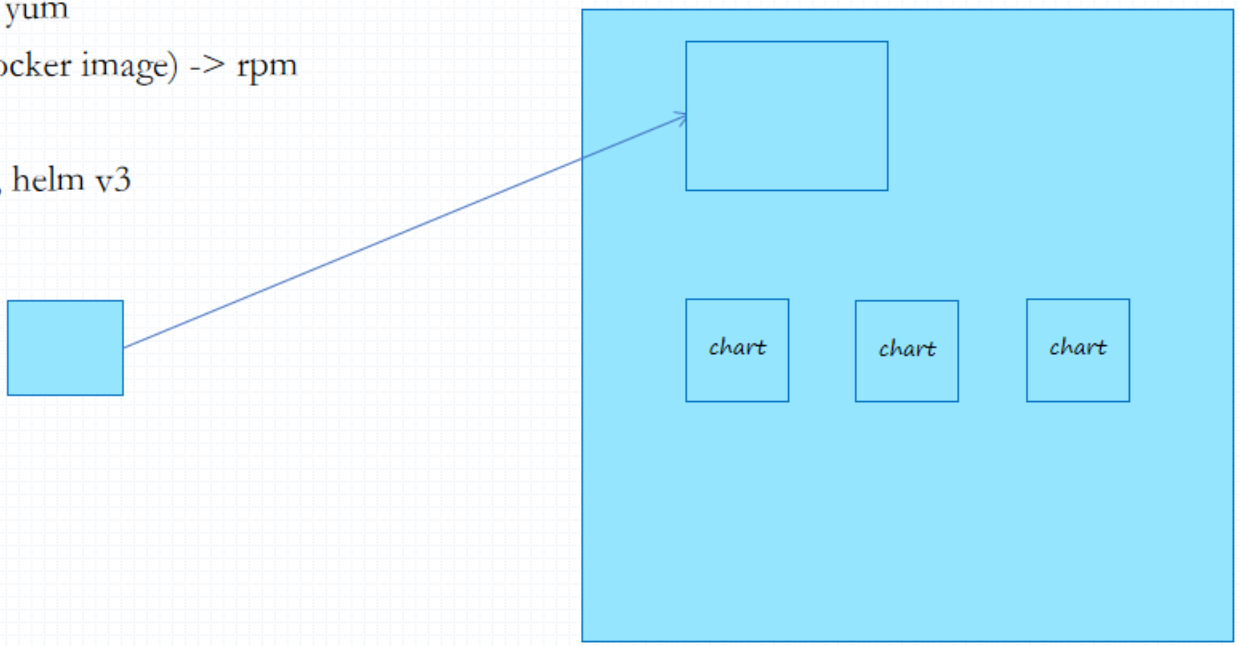
- 组织一个生态圈内多个组件为一体，统一进行管理
  - deb, rpm -> apt, yum --> 包仓库
  - kubectl, kustomize声明式应用管理（v1.14）
  - helm(头盔), chart（图表）--> Chart Hub
    - Prometheus-Server
      - ✓ controller, service, configmap, secret, template
      - ✓ helm -> operator -> manifests (CRD)



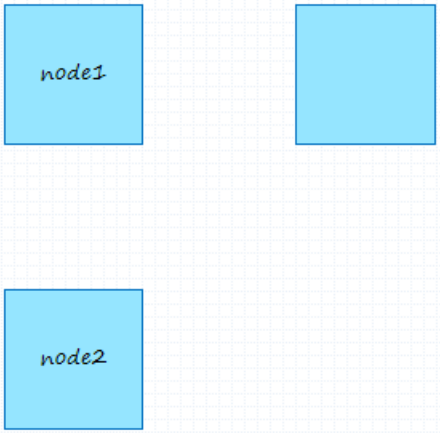


- kustomize， HUB
- helm -> yum
- chart (docker image) -> rpm

- helm v2, helm v3



kubelet, cAdvisor, Heapster



- 核心指标: metrics-server, mtrics
- 自定义指标: prometheus,

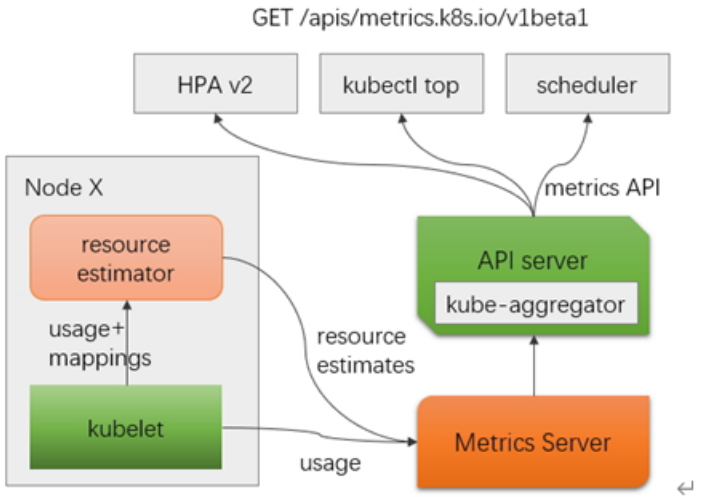


图 15-3 资源指标 API 系统组件

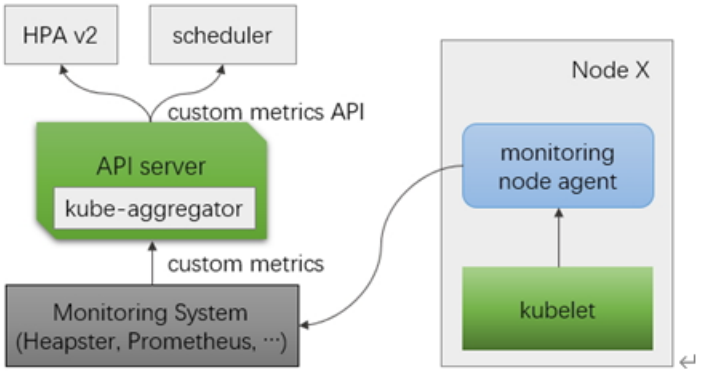
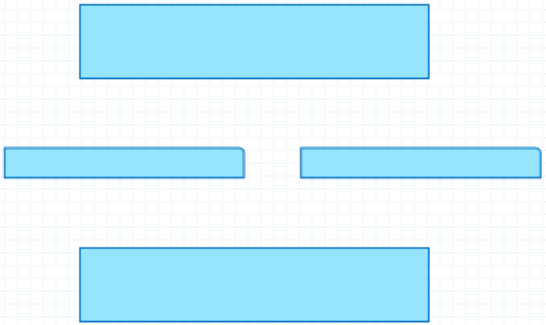


图 15-4 自定义指标 API 系统组件

◆prometheus的指标格式，与k8s的指标格式不兼容；



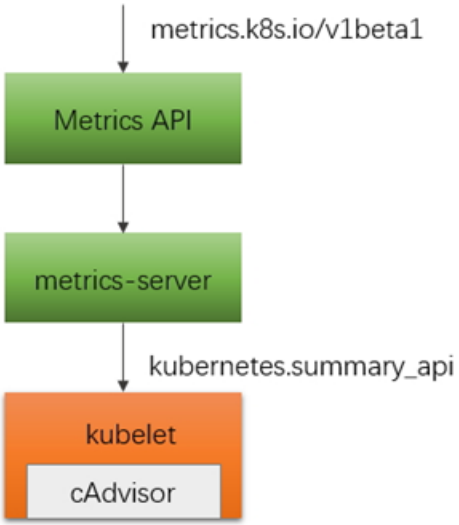


图 15-5 资源指标 API 架构简图

自定义api server, APIService

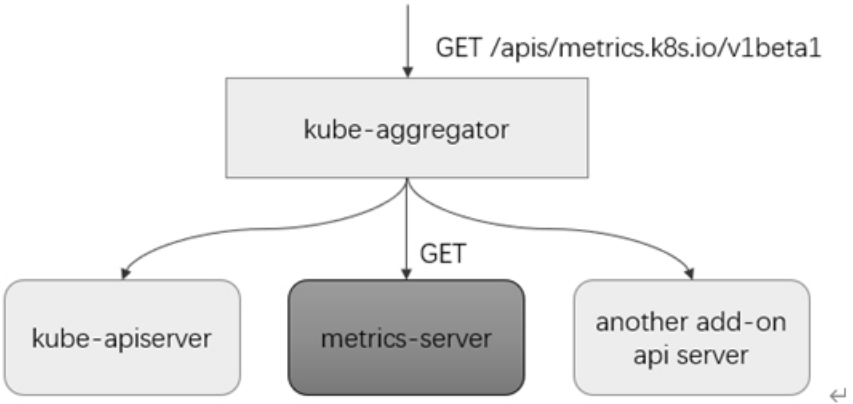


图 15-6 聚合 metrics-server 于主 API server 上

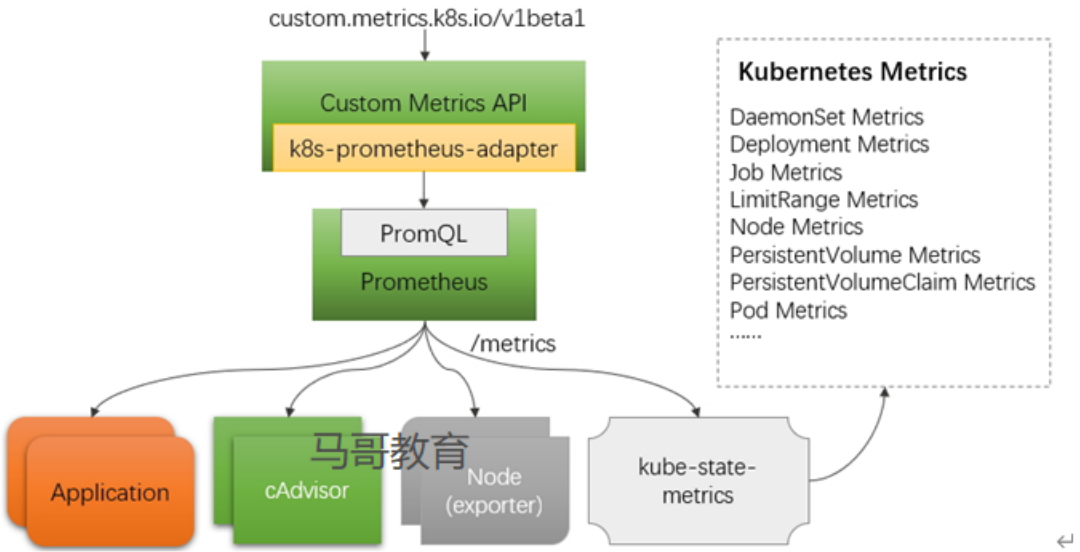

custom.metrics.k8s.io/v1beta1
Custom Metrics API
k8s-prometheus-adapter
PromQL
Prometheus
/metrics
Application
cAdvisor
Node
(exporter)
kube-state-
metrics
Kubernetes Metrics
DaemonSet Metrics
Deployment Metrics
Job Metrics
LimitRange Metrics
Node Metrics
PersistentVolume Metrics
PersistentVolumeClaim Metrics
Pod Metrics
.....
马哥教育

图 15-8 自定义指标

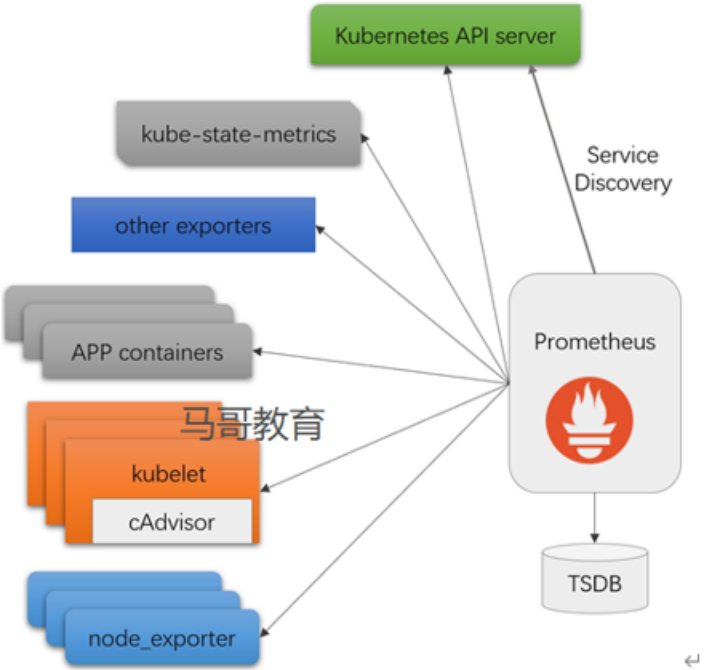


图 15-12Kubernetes 中的 Prometheus 数据源

- Istio, chart, chart hub
- helm install istio

关于马哥教育

- http://www.magedu.com
- https://github.com/ikubernetes

Thank You!

讲师：马永亮（马哥）
http://www.magedu.com