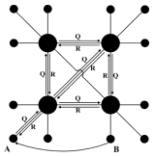
DATAGURU 炼数成金

Leader-us系列



从Docker到Kubernetes 第9周

法律声明

【声明】本视频和幻灯片为炼数成金网络课程的教学资料，所有资料只能在课程内使用，不得在课程以外范围散播，违者将可能被追究法律和经济责任。

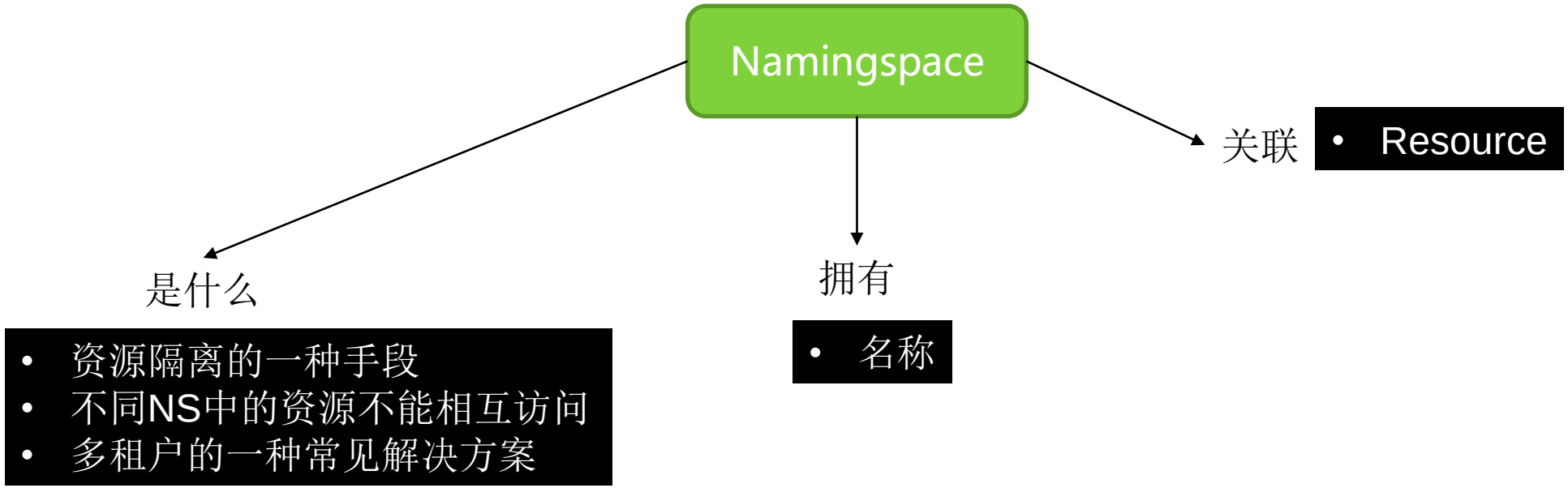
课程详情访问炼数成金培训网站

http://edu.dataguru.cn

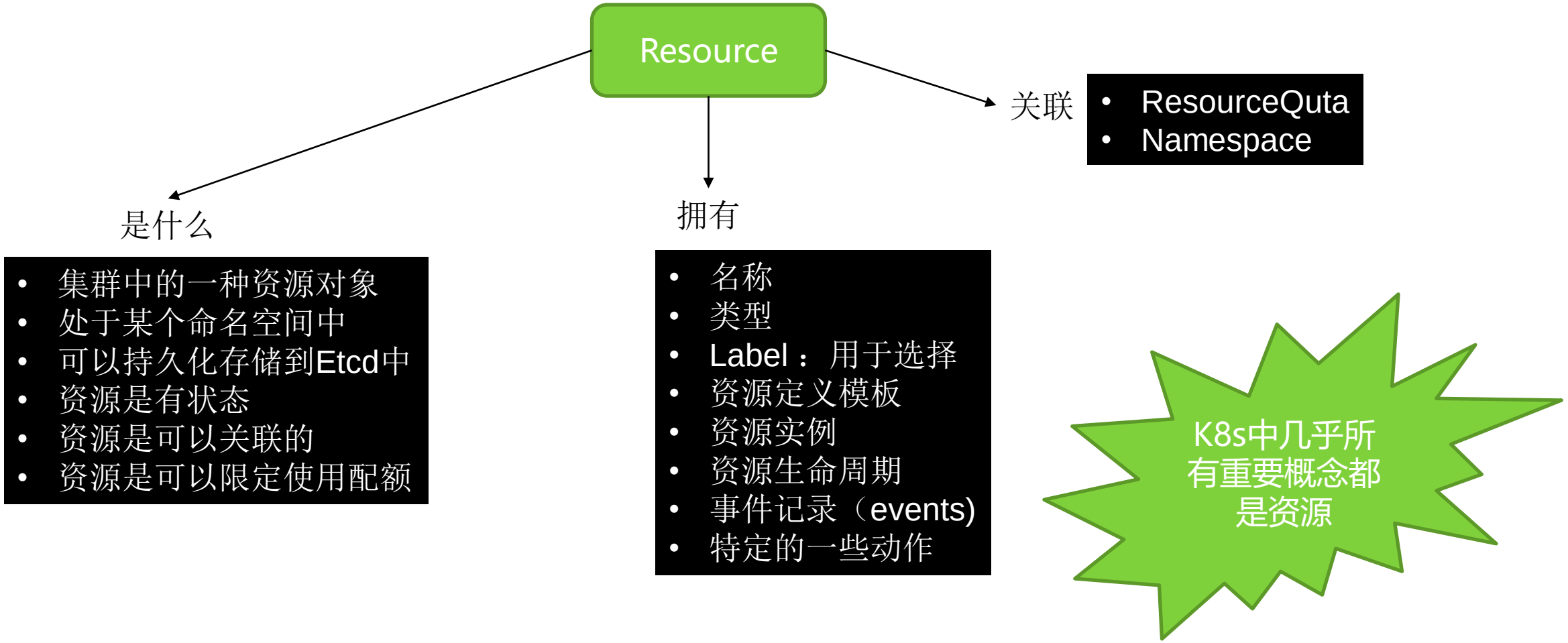
第九课kubernetes架构深度解析

- Kubernetes重要概念
- Kubernetes架构及原理
- Kubernetes网络机制

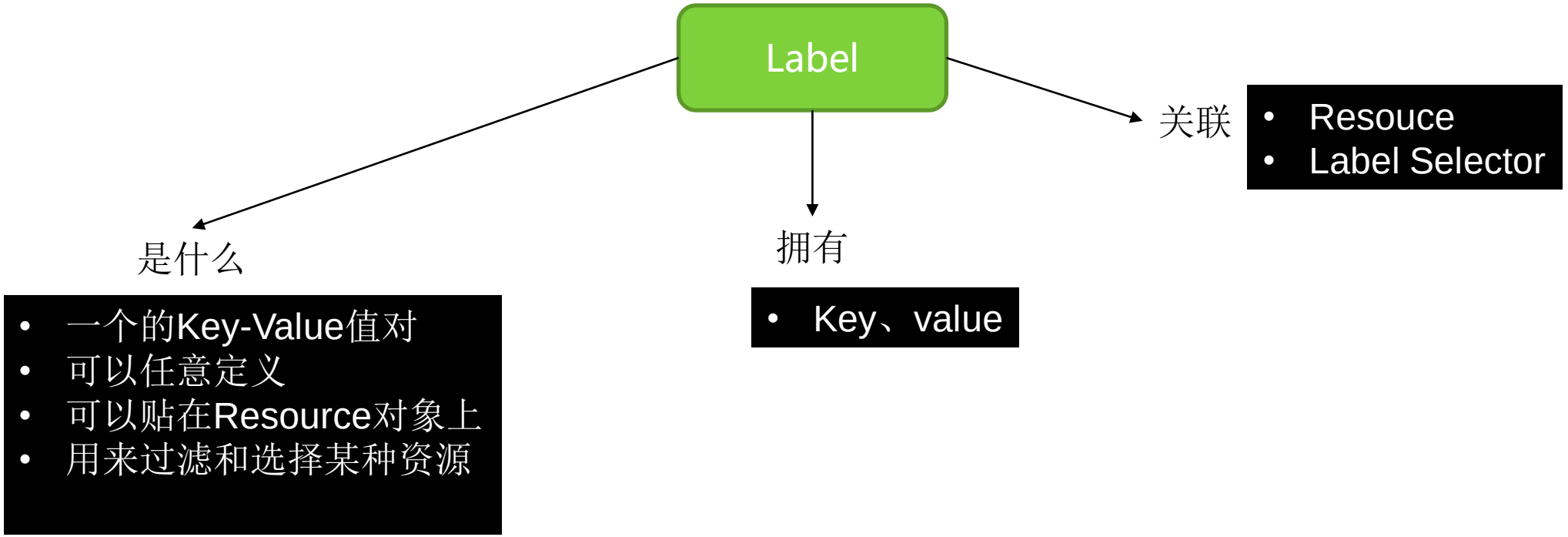
Kubernetes重要概念



Kubernetes重要概念



Kubernetes重要概念



Kubernetes重要概念

Label

```
"labels": {
  "key1" : "value1",
  "key2" : "value2"
}
```

Label Selector

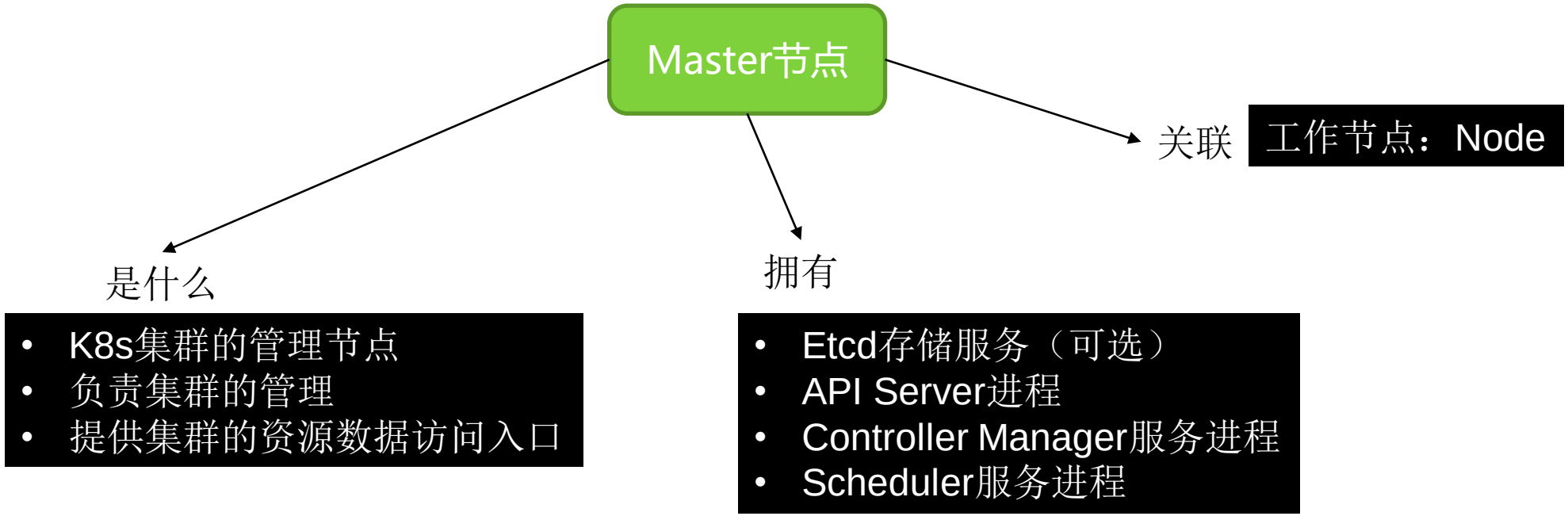
name in (redis-master, redis-slave)

选择所有包含Label中（key="name"， value="redis-master"或"redis-slave"）的对象。

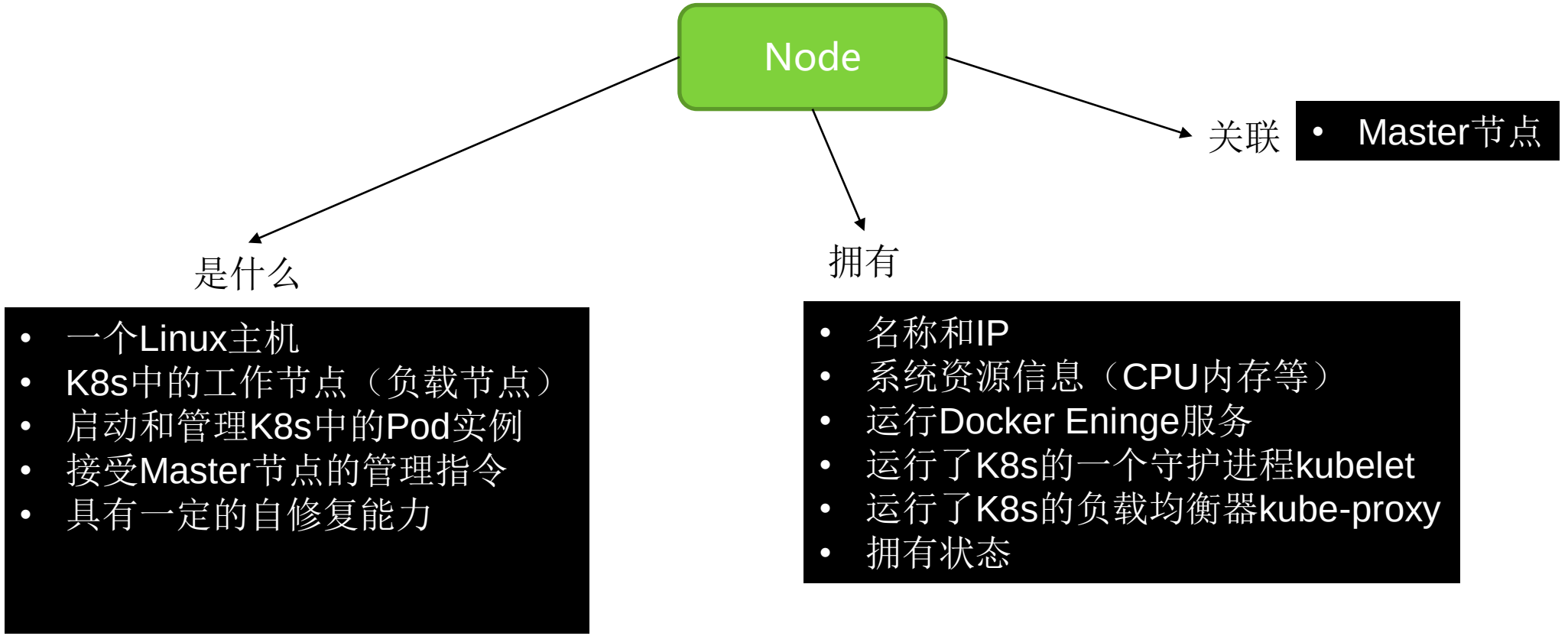
```
name=redis-slave,env!=production
```

```
apiVersion: v1
kind: ReplicationController
metadata:
  name: redis-slave
  labels:
    name: redis-slave
spec:
  replicas: 2
  selector:
    name: redis-slave
  template:
    metadata:
      labels:
        name: redis-slave
    spec:
      containers:
      - name: slave
        image: redis-slave
        ports:
        - containerPort: 6379
```

Kubernetes重要概念



Kubernetes重要概念



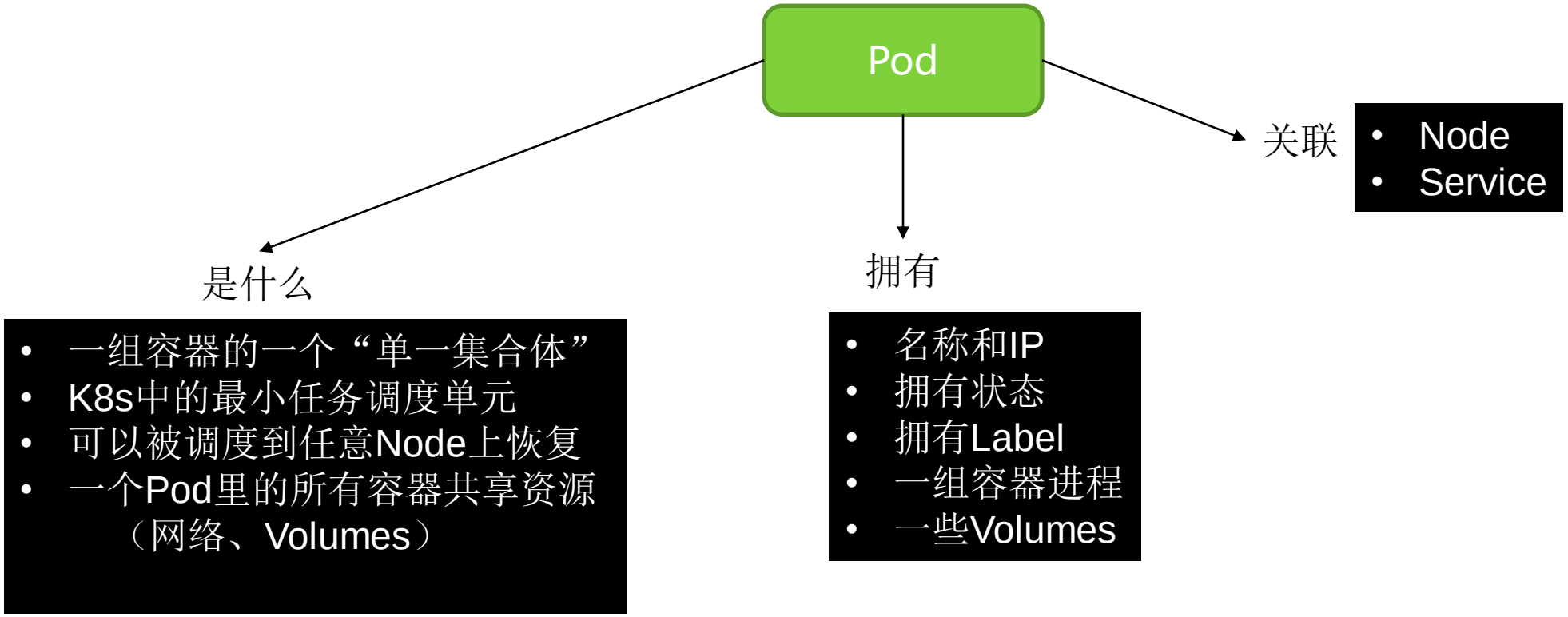
Kubernetes重要概念

Node

```
kubectl describe node kubernetes-minion1
Name:                   kubernetes-minion1
Labels:                 kubernetes.io/hostname=kubernetes-minion1
CreationTimestamp:      Tue, 04 Aug 2015 14:34:22 +0800
Conditions:
  Type          Status  LastHeartbeatTime                       LastTransitionTime                      Reason
Message
  Ready         True    Thu, 13 Aug 2015 12:12:03 +0800         Tue, 11 Aug 2015 09:37:17 +0800         kubelet is
posting ready status
Addresses:      192.168.1.129
```

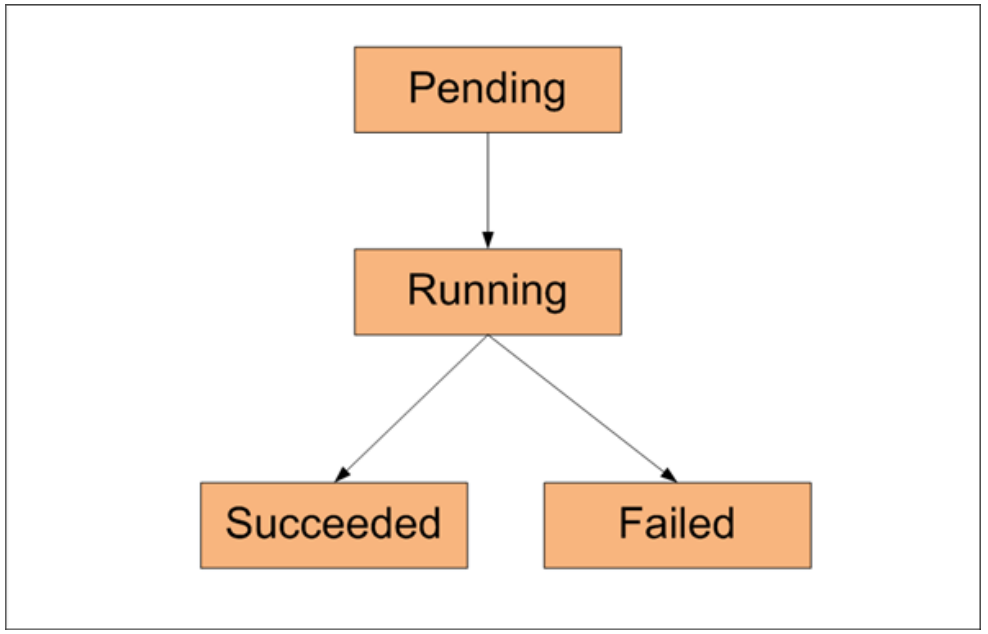
Capacity:
 cpu: 2
 memory: 1870516Ki
 pods: 40
Version:
 Kernel Version: 3.10.0-229.el7.x86_64
 OS Image: Red Hat Enterprise Linux Server 7.1 （Maipo）
 Container Runtime Version: docker://1.6.2.el7
 Kubelet Version: v1.0.0
 Kube-Proxy Version: v1.0.0
ExternalID: kubernetes-minion1
Pods: (6 in total)
  Namespace Name
  default redis-slave-4na2n
  default redis-slave-92u3k
No events.

Kubernetes重要概念



Kubernetes重要概念

Pod

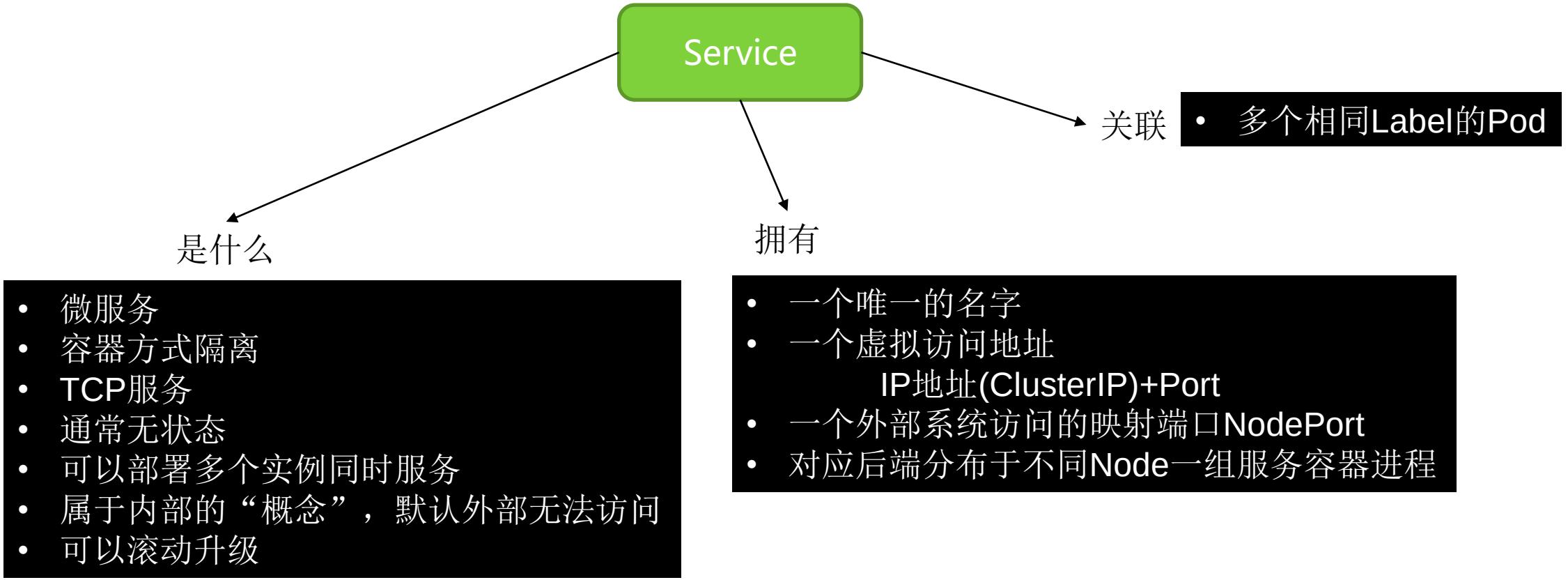


Pod状态

```
apiVersion: v1
kind: Pod
metadata:
  name: redis-slave
  labels:
    name: redis-slave
spec:
  containers:
  - name: slave
    image: kubeguide/guestbook-redis-slave
    env:
    - name: GET_HOSTS_FROM
      value: env
    ports:
    - containerPort: 6379
```

```
$ kubectl get endpoints
NAME           ENDPOINTS
redis-master   172.16.42.6:6379
```

Kubernetes重要概念



Kubernetes重要概念

Service

```yaml
apiVersion: v1
kind: Service
metadata:
  name: redis-slave
  labels:
    name: redis-slave
spec:
  ports:
  - port: 6379
  selector:
    name: redis-slave
```

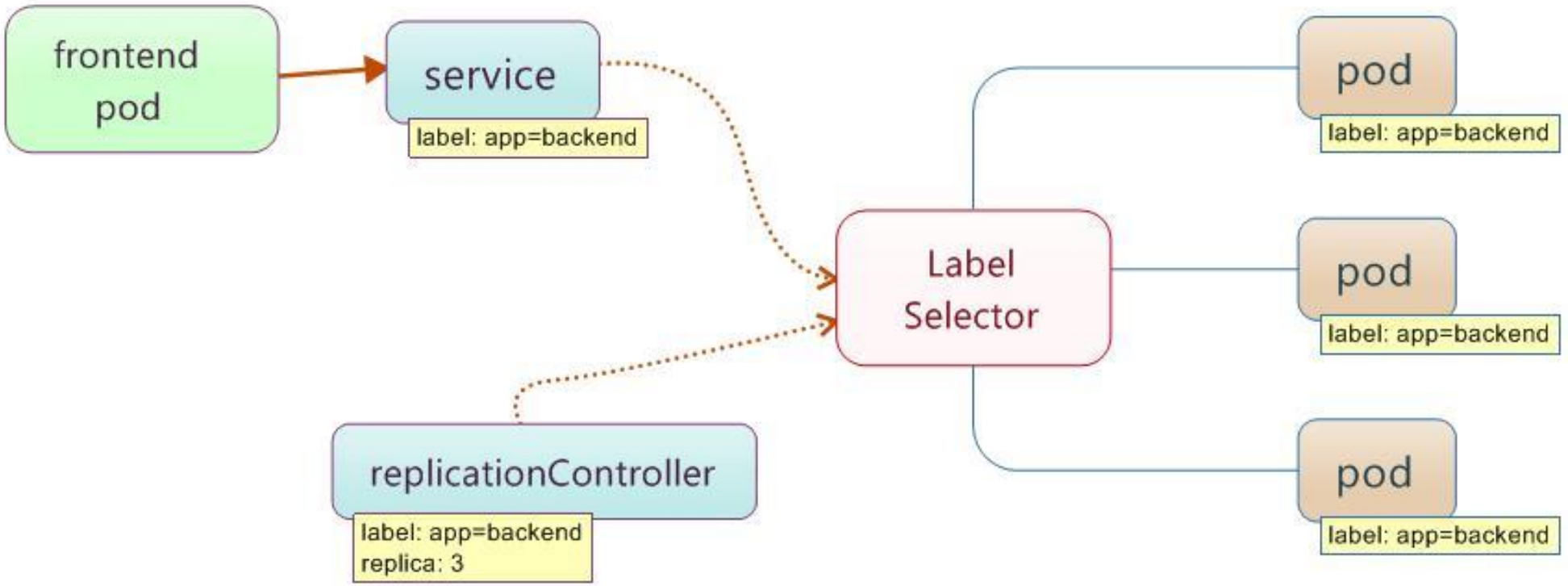
Service定义

```
{
    "kind": "Service",
    "apiVersion": "v1",
    "metadata": {
        "name": "my-service"
    },
    "spec": {
        "selector": {
            "app": "MyApp"
        },
        "ports": [
            {
                "name": "http",
                "protocol": "TCP",
                "port": 80,
                "targetPort": 9376
            },
            {
                "name": "https",
                "protocol": "TCP",
                "port": 443,
                "targetPort": 9377
            }
        ]
    }
}
```

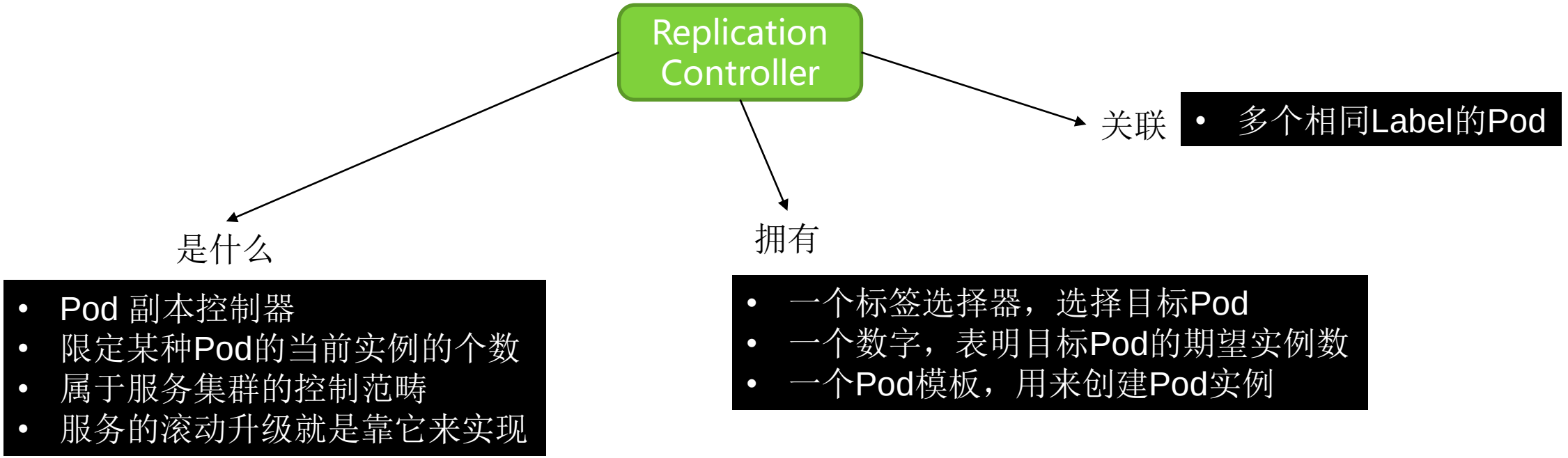
Service实例

Kubernetes重要概念

Service



Kubernetes重要概念



Kubernetes重要概念

Replication Controller

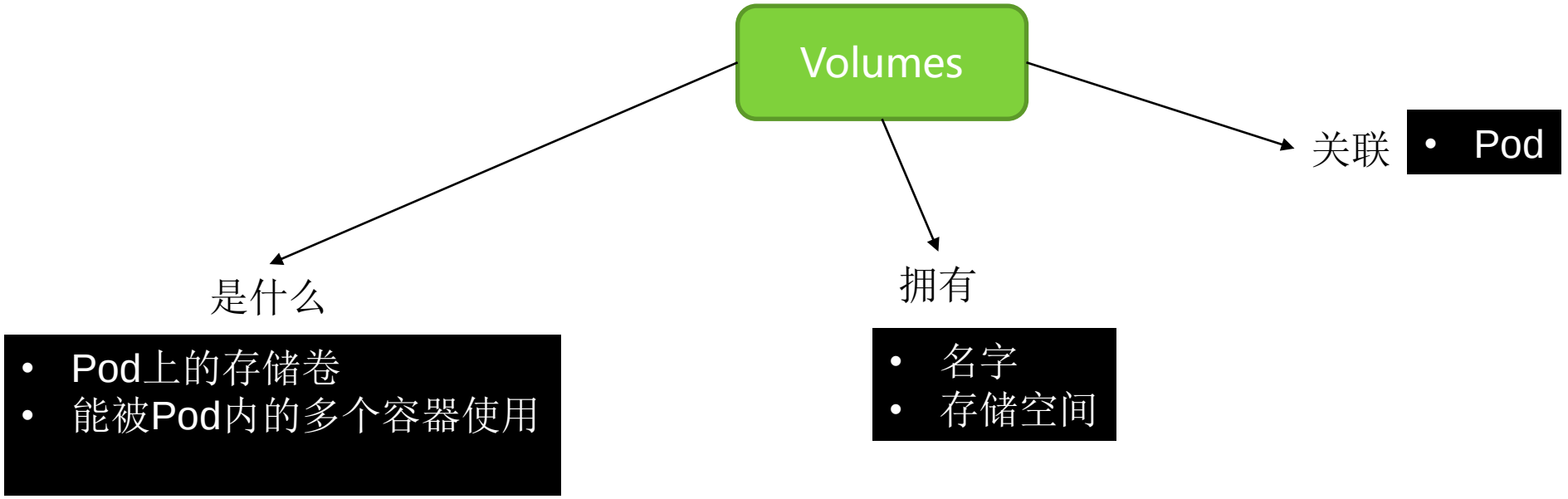
```
apiVersion: v1
kind: ReplicationController
metadata:
  name: redis-slave
  labels:
    name: redis-slave
spec:
  replicas: 2
  selector:
    name: redis-slave
  template:
    metadata:
      labels:
        name: redis-slave
    spec:
      containers:
      - name: slave
        image: kubeguide/guestbook-redis-slave
        env:
        - name: GET_HOSTS_FROM
          value: env
        ports:
        - containerPort: 6379
```

Pod实例数

标签选择器

Pod模板

Kubernetes重要概念



Volumes（存储卷）是Pod中能够被多个容器访问的共享目录。Kubernetes的Volumes概念与Docker的Volumes比较类似，但并不完全相同。Kubernetes中的Volumes与Pod生命周期相同，而不与容器的生命周期相关。当容器终止或者重启时，Volumes中的数据也不会丢失。另外，Kubernetes支持多种类型的Volumes，并且一个Pod可以同时使用任意多个Volumes。

Kubernetes重要概念

Volumes

1. **emptyDir**：一个EmptyDir Volume是在Pod分配到Node时创建的。从它的名称就可以看出，它的初始内容为空。同一个Pod中所有容器可以读和写EmptyDir中的相同文件。当Pod从Node上移除时，EmptyDir中的数据也会永久删除。

   EmptyDir的一些用途如下：

- 临时空间，例如用于某些应用程序运行时所需的临时目录，且无需永久保留；
- 长时间任务的中间过程CheckPoint临时保存目录；
- 一个容器需要从另一个容器中获取数据的目录（多容器共享目录）。

1. **hostPath**：在Pod上挂载宿主机上的文件或目录。

   hostPath通常可以用于：

- 容器应用程序生成的日志文件需要永久保存，可以使用宿主机的高速文件系统进行存储；
- 需要访问宿主机上Docker引擎内部数据结构的容器应用，可以通过定义hostPath为宿主机/var/lib/docker目录使得容器内部应用可以直接访问Docker的文件系统。

  当使用这种类型的Volume时，需要注意：

- 在不同Node上具有相同配置的Pod可能会因为宿主机上的目录和文件不同而导致对Volume上目录和文件的访问结果不一致；

Kubernetes重要概念

Volumes

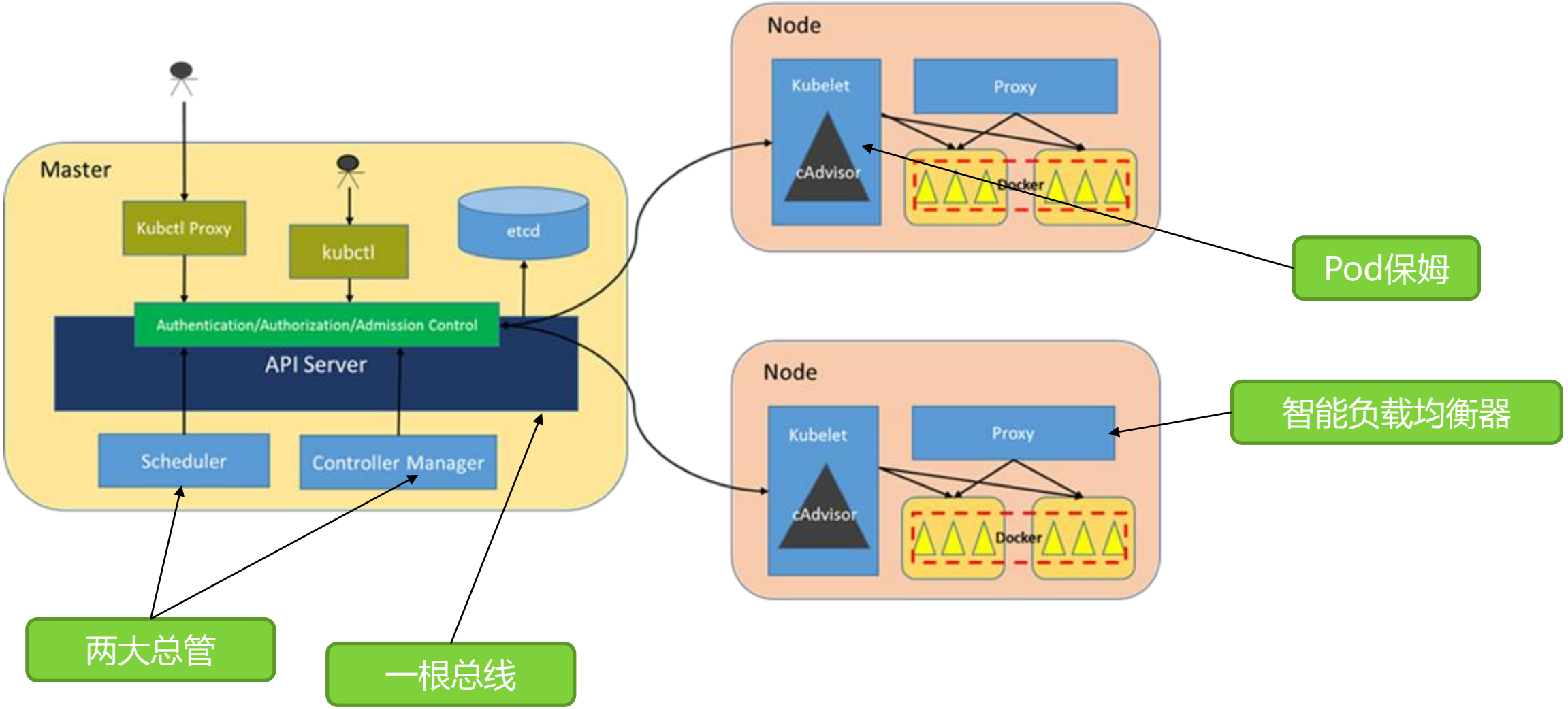
```
apiVersion: v1
kind: ReplicationController
metadata:
  name: redis-master
  labels:
    name: redis-master
spec:
  replicas: 1
  selector:
    name: redis-master
  template:
    metadata:
      labels:
        name: redis-master
    spec:
      volumes:
      - name: "persistent-storage"
        hostPath:
          path: "/data"
      containers:
      - name: master
        image: kubeguide/redis-master
        ports:
        - containerPort: 6379
        volumeMounts:
        - name: "persistent-storage"
          mountPath: "/data"
```

宿主机路径

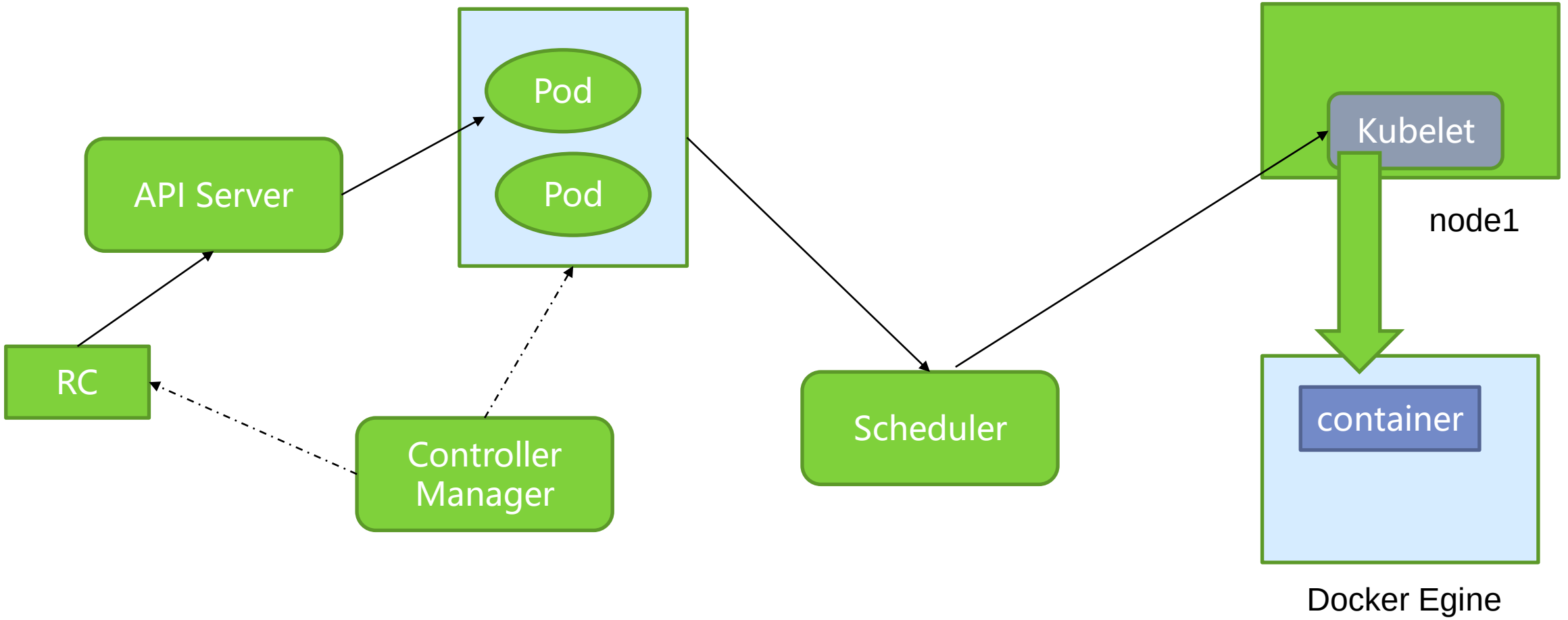
容器路径

```
- name: web
      image: nginx
      ports:
        - name: web
          containerPort: 80
      volumeMounts:
        - name: nfs
          mountPath: "/usr/share/nginx/html"
  volumes:
    - name: nfs
      nfs:
        # 改为你的NFS服务器地址
        server: nfs-server.localhost
        path: "/"
```

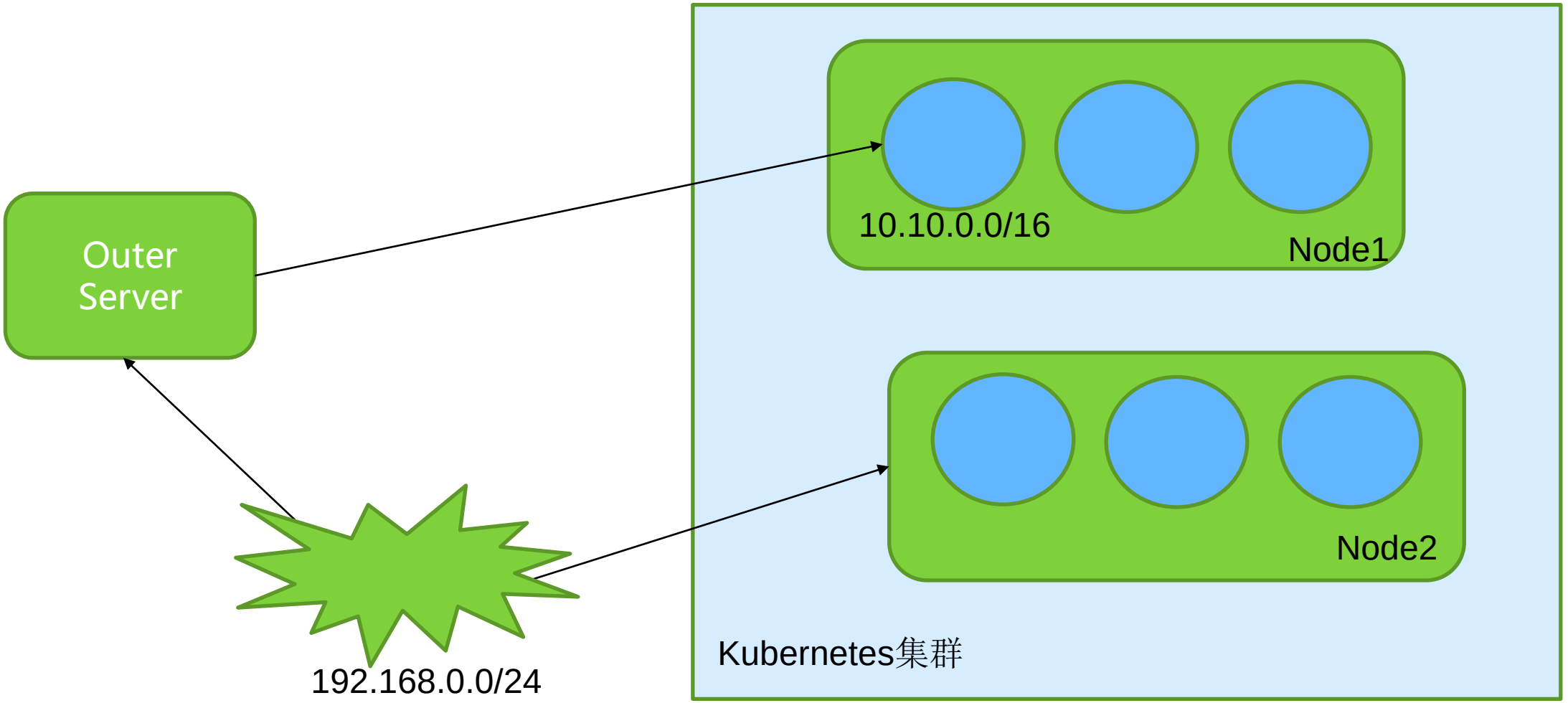
Kuberetes架构及原理



Kuberetes架构及原理——运行机制



Kuberetes架构及原理——网络机制



Kuberetes架构及原理——网络机制

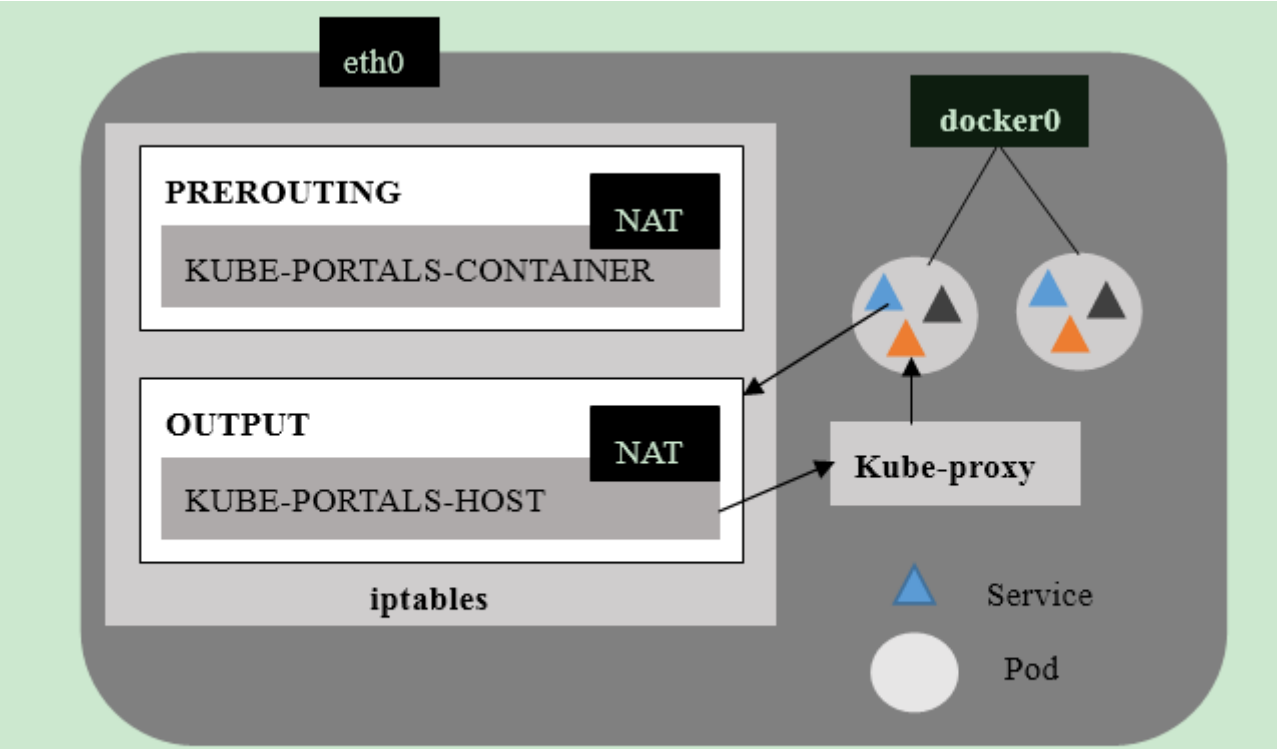


图 7.10　Service Portal 方式访问 Service 的流量示意图

Kuberetes架构及原理——网络机制

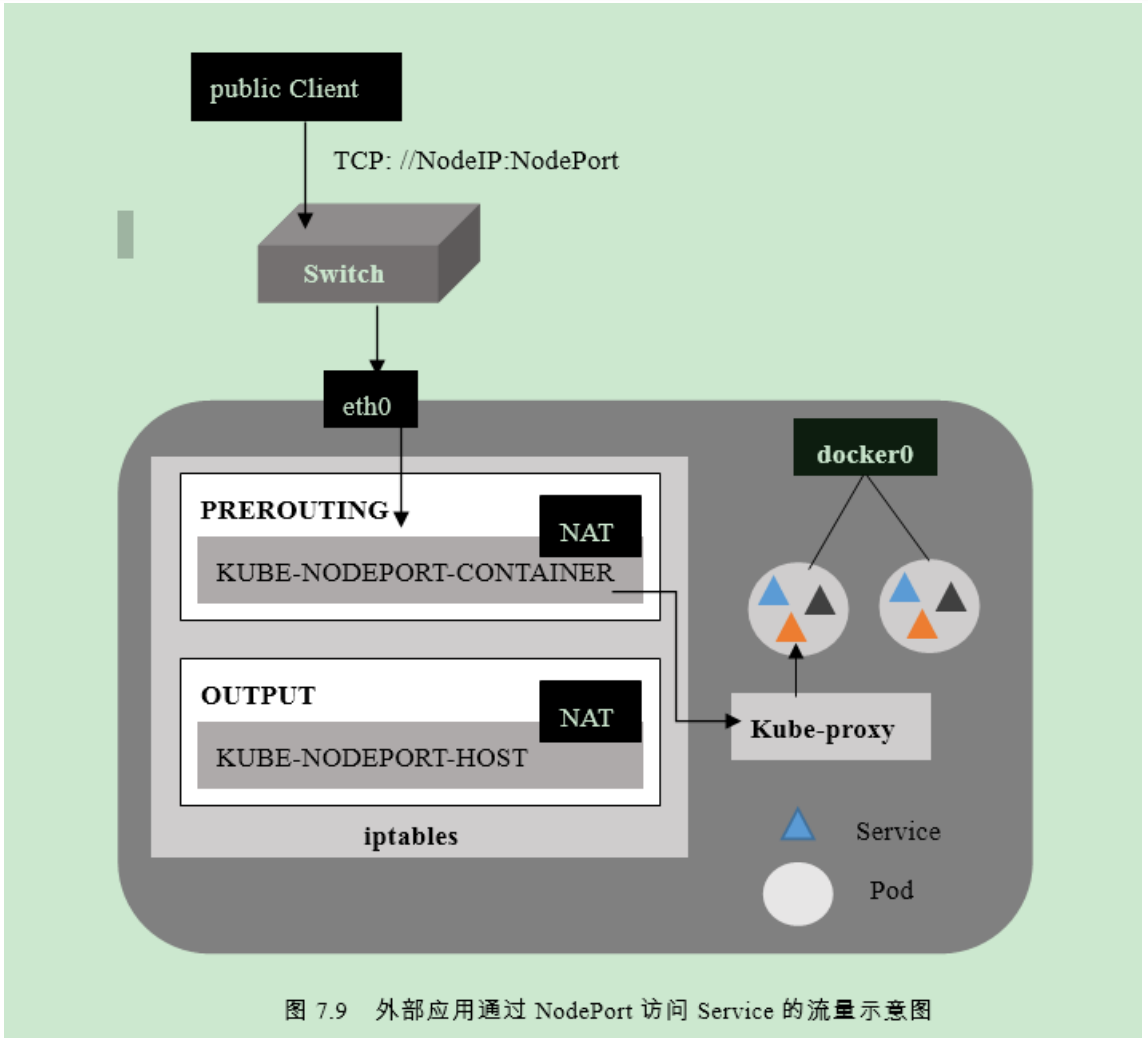


图 7.9　外部应用通过 NodePort 访问 Service 的流量示意图

Thanks

FAQ时间