kubernetes入门与实践

kubernetes概述

Kubernetes概述

- Kubernetes又称作k8s，是**Google**在2014年发布的一个开源项目。
- 最初Google开发了一个叫**Borg**的系统（现在命名为Omega），来调度近20多亿个容器。在积累了数十年的经验后，Google决定重写这个容器管理系统，并贡献给开源社区，而这个系统就是Kubernetes。它也是**Omega**的开源版本。
- 从2014年第一个版本发布以来，迅速得到了开源社区的追捧，目前，k8s已经成为了发展最快、市场占有率最高的容器编排引擎产品。
- Kubernetes中文社区丨中文文档 https://www.kubernetes.org.cn/k8s

安装minikube

安装minikube

- 设置阿里云镜像

  vim /etc/yum.repos.d/kubernetes.repo

```
[kubernetes]
name=Kubernetes
baseurl=http://mirrors.aliyun.com/kubernetes/yum/repos/kubernetes-el7-x86_64/
enabled=1
gpgcheck=0
```

- 安装minikube

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

- 安装kubectl

  curl -LO https://dl.k8s.io/release/v1.20.0/bin/linux/amd64/kubectl
  sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl

- 安装conntrack

  yum install conntrack

我们先操作一波

- 启动minikube
  minikube start --vm-driver=none --image-mirror-country='cn'
- 停止minikube
  minikube stop

- 查看节点
  kubectl get nodes
- 创建nginx的deployment
  kubectl create deployment my-nginx --image nginx:latest
- 查看所有deployment
  kubectl get deployment
- 查看所有pod
  kubectl get pods
- 查看所有pod信息以及ip和port
  kubectl get pods -o wide
- 查看所有service
  kubectl get services

- 查询所有命名空间
  kubectl get namespace
- 将副本数修改为3个
  kubectl scale deployments/my-nginx --replicas=3

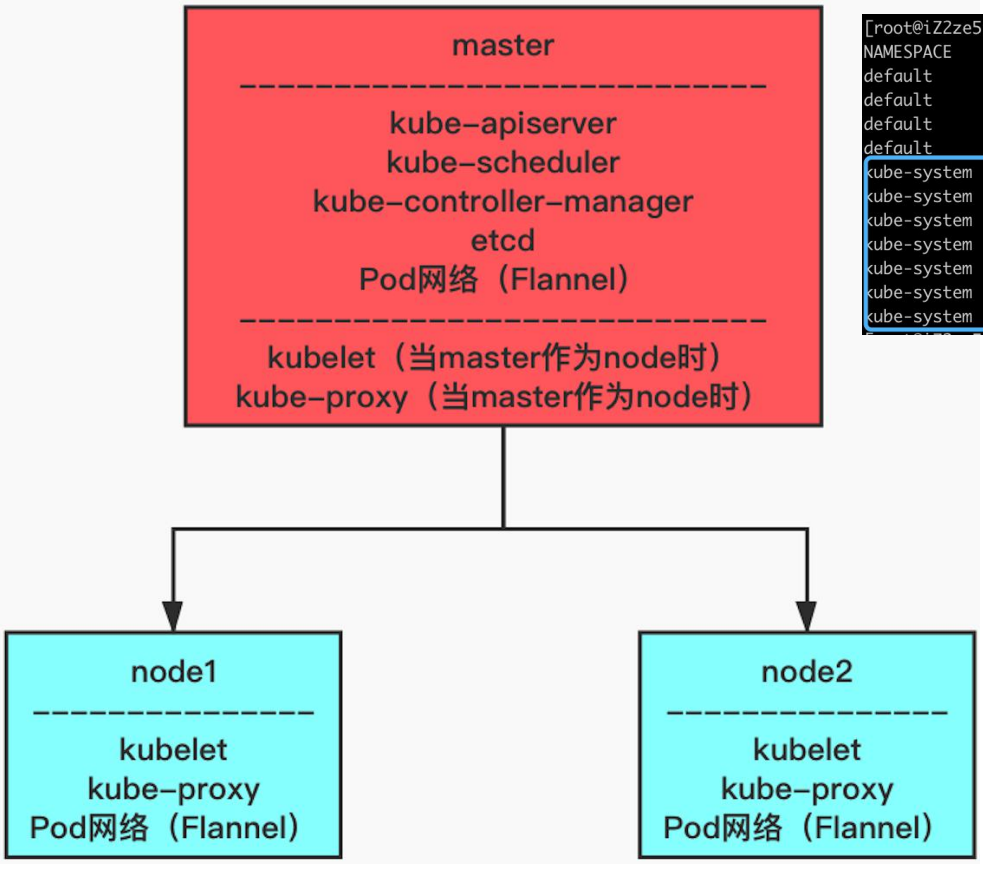
k8s重要概念介绍

重要概念

- Cluster——是计算、存储和网络资源的集合
- Master——Cluster的大脑，决定将应用放在哪里运行
- Node——职责是运行容器应用
- Pod——k8s的最小工作单元，包含1orN个容器。
- Controller——k8s通过它来管理Pod
  包含：Deployment、ReplicaSet、DaemonSet、StatefulSet、Job
- Service——为Pod提供了负载均衡、固定的IP和Port
- Namespace——解决同一个Cluster中，如何区别分开Controller、Pod等资源的问题

kubernetes架构

kubernetes架构图




```
[root@iZ2ze5ffbqqbeaygcx7o4xZ ~]# kubectl get pod --all-namespaces
NAMESPACE     NAME                                                READY   STATUS      RESTARTS   AGE
default       myjob-9g8rz                                         0/1     Completed   0          5d10h
default       myjob-ghdhr                                         0/1     Completed   0          5d10h
default       nginx-deployment-7f4fc68488-lzbb7                   1/1     Running     0          6d4h
default       nginx-deployment-7f4fc68488-msmq4                   1/1     Running     0          6d4h
kube-system   coredns-54d67798b7-nwkfg                            1/1     Running     0          6d4h
kube-system   etcd-iz2ze5ffbqqbeaygcx7o4xz                        1/1     Running     0          6d4h
kube-system   kube-apiserver-iz2ze5ffbqqbeaygcx7o4xz              1/1     Running     0          6d4h
kube-system   kube-controller-manager-iz2ze5ffbqqbeaygcx7o4xz     1/1     Running     0          6d4h
kube-system   kube-proxy-c79lt                                    1/1     Running     0          6d4h
kube-system   kube-scheduler-iz2ze5ffbqqbeaygcx7o4xz              1/1     Running     0          6d4h
kube-system   storage-provisioner                                 1/1     Running     0          6d4h
```

- Kubernetes的系统组件都被放到kube-system的namespace中。
- kubelet是唯一没有以容器形式运行的Kubernetes组件。

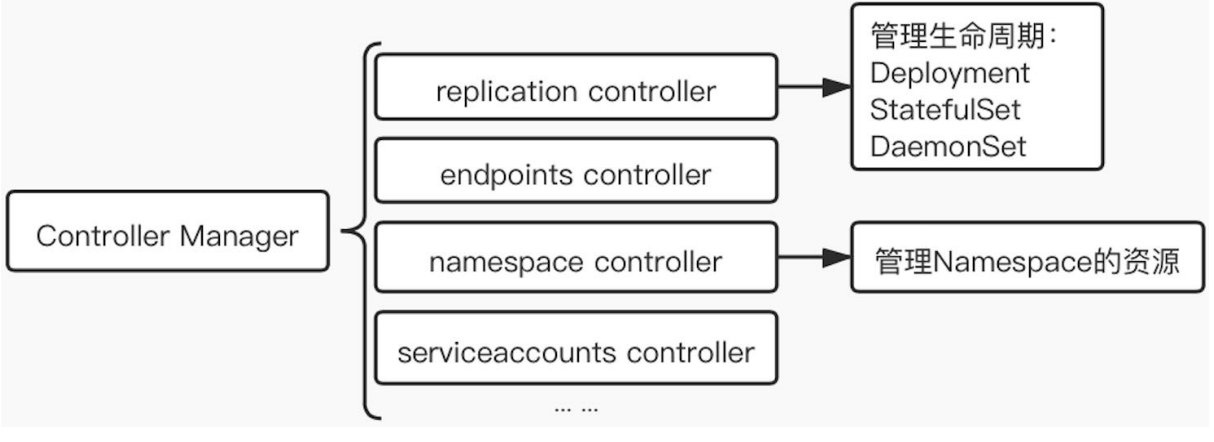
- 当我们执行部署应用并指定两个副本的时候，执行流程如下所示：
  1> Kuberctl发送部署请求到API Server。
  2> API Server通知Controller Manager创建一个deployment资源。
  3> Scheduler执行调度任务，将两个副本Pod分发到node1和node2上。
  4> node1和node2上的kubectl在各自的节点上创建并运行Pod。

- k8s架构中，主要是由Master和Node组成的。

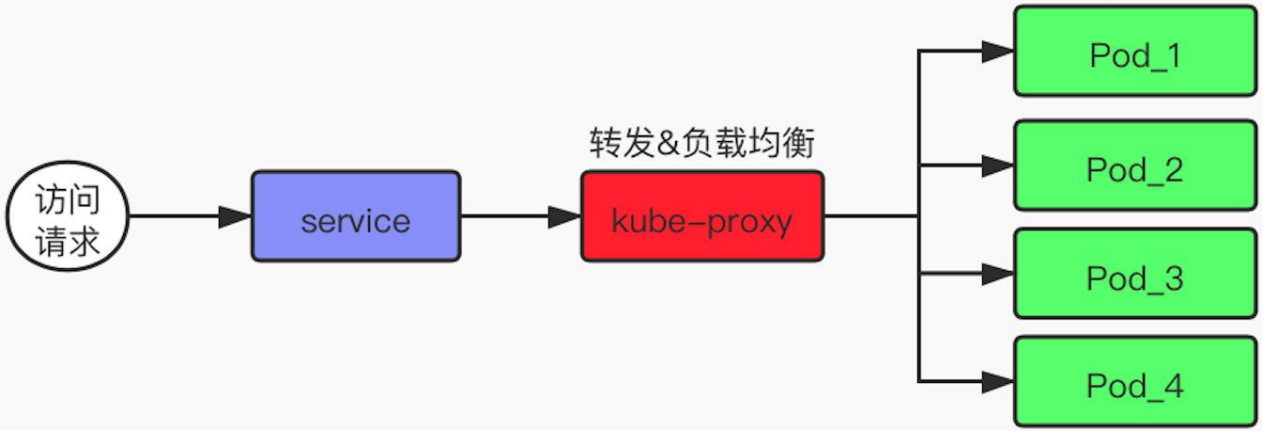
下面我们来针对这两部分进行详细的介绍。

Master

- API-Server
- Scheduler
- Controller Manager
- etcd
- Pod网络



- kubelet
- kube-proxy
- Pod网络



Deployment

创建资源的方式

- 方式一
  用kubectl命令直接创建。
  比如：kubectl run nginx-deployment--image=nginx:1.7.9--replicas=2
  在命令行中通过参数指定资源的属性。（但是，在K8S v1.18.0以后，—replicas已弃用 ,推荐用 kubectl apply 创建 pods)

- 方式二
  通过配置文件和kubectl apply创建。
  步骤：
  1> 编写yml配置文件。（下一页有书写样例，nginx.yml)
  2> 执行命令：kubectl apply -f /home/muse/nginx.yml

nginx.yml配置文件

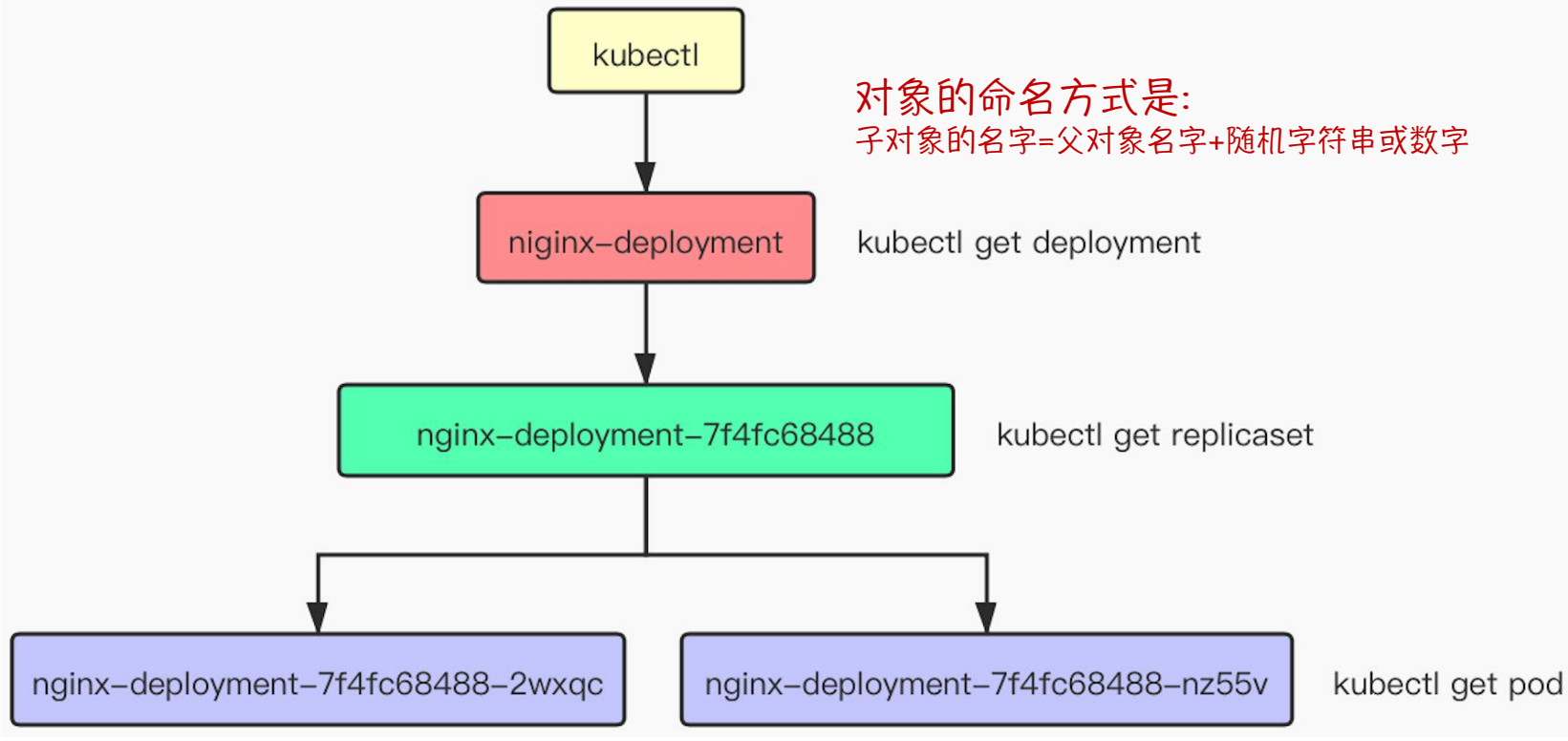
```
[root@iZ2ze5ffbqqbeaygcx7o4xZ muse]# cat nginx.yml
# API 版本号
apiVersion: apps/v1
# 类型，如：Pod/ReplicationController/Deployment/Service/Ingress
kind: Deployment
metadata:
  # Kind 的名称
  name: nginx-deployment
spec:
  selector:
    matchLabels:
      # 容器标签的名字，发布 Service 时，selector 需要和这里对应
      app: nginx
  # 部署的实例数量
  replicas: 2
  template:
    metadata:
      labels:
        app: nginx
    spec:
      # 配置容器，数组类型，说明可以配置多个容器
      containers:
      # 容器名称
      - name: nginx
        # 容器镜像
        image: nginx:1.17
        # 只有镜像不存在时，才会进行镜像拉取
        imagePullPolicy: IfNotPresent
        ports:
        # Pod 端口
        - containerPort: 80
```

- replicas: 2
  部署的副本实例数量，默认为1

- metadata:
  metadata定义Pod的元数据，至少要定义一个label。label的key和value可以任意指定

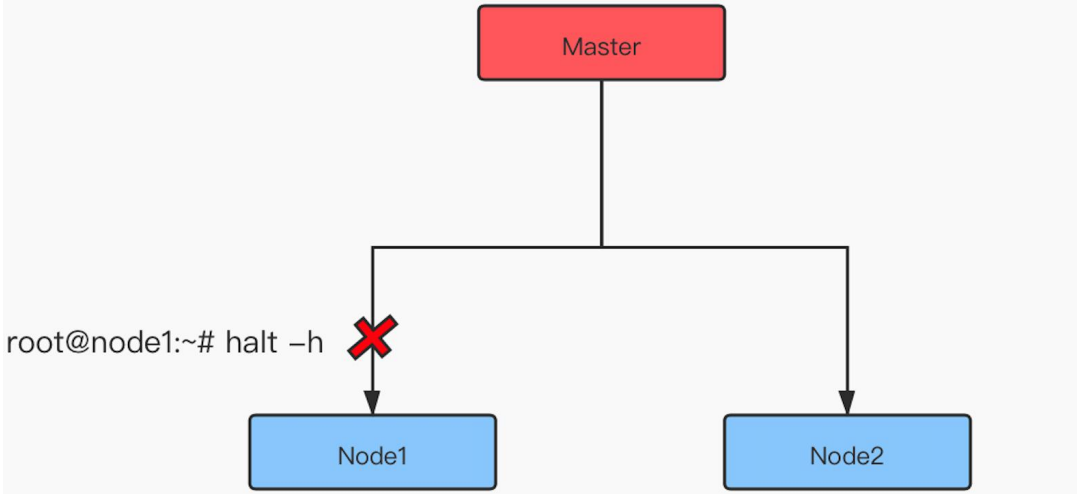
- spec:
  描述Pod的规格，此部分定义Pod中每一个容器的属性，name和image是必需的

构建过程解析



用户通过kubectl——>创建Deployment——>创建ReplicaSet——>创建Pod

Failover



最初状态	Pod1 STATUS=Running Pod2 STATUS=Running	Pod3 STATUS=Running
Node1节点异常	Pod1 STATUS=Unknown Pod2 STATUS=Unknown	Pod3 STATUS=Running Pod4 STATUS=Running Pod5 STATUS=Running
Node1节点恢复	两个Unknown Pod被删除	Pod3 STATUS=Running Pod4 STATUS=Running Pod5 STATUS=Running

- 设置了pod数为3个
- 当Node1异常的时候，会在Node2上面生成新的Pod来维护总数为3个pod
- 当Node1恢复正常的时候，新创建的pod也依然会在Node2上，并不会做迁移动作。

label

- 默认配置下，Scheduler会将Pod调度到所有可用的Node。不过有些情况我们可以通过lable将Pod部署到指定的Node，比如将有大量磁盘I/O的Pod部署到配置了SSD的Node；或者Pod需要GPU，需要运行在配置了GPU的节点上。

- 给k8s-node1添加标签——disktype=ssd

  kubectl label node k8s-node1disktype=ssd

- 修改nginx.yml配置文件，指定nodeSelector为上一步新建的label。

  ```
  nodeSelector:
      disktype: ssd
  ```

- 重新部署Deployment

  kubectl apply -f nginx.yml

- 查看节点的标签信息

  kubectl get node --show-labels

删除deployment

- 假设现在配置的是2个pod数。那么如果我们只是删除其中的一个pod，依然会被deployment根据配置，再补充为2个pod。
- 当我们删除掉deployment的时候，pod也会随之自动被删除。
- 删除pod
  kubectl delete pod nginx-deployment-7f4fc68488-5v4m7
- 删除deployment
  kubectl delete deployment nginx-deployment

DaemonSet

DaemonSet

- Deployment部署的副本Pod会分布在各个Node上，每个Node都可能运行好几个副本。DaemonSet的不同之处在于：每个Node上最多只能运行一个副本。

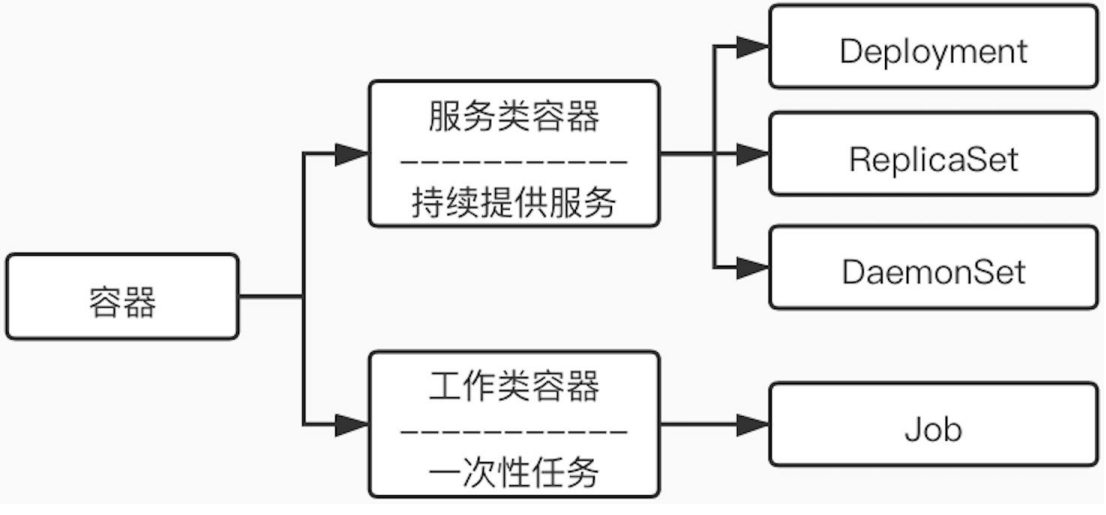
- DaemonSet的典型应用场景
  (1) 在每个节点上运行存储Daemon，比如glusterd或ceph。
  (2) 在每个节点上运行日志收集Daemon，比如flunentd或logstash。
  (3) 在每个节点上运行监控Daemon，比如Prometheus Node Exporter或collectd。

- 查看k8s自己就用DaemonSet运行系统组件

```
[root@iZ2ze5ffbqqbeaygcx7o4xZ muse]# kubectl get daemonset --namespace=kube-system
NAME         DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR            AGE
kube-proxy   1         1         1       1            1           kubernetes.io/os=linux   6d7h
```

Job

Job



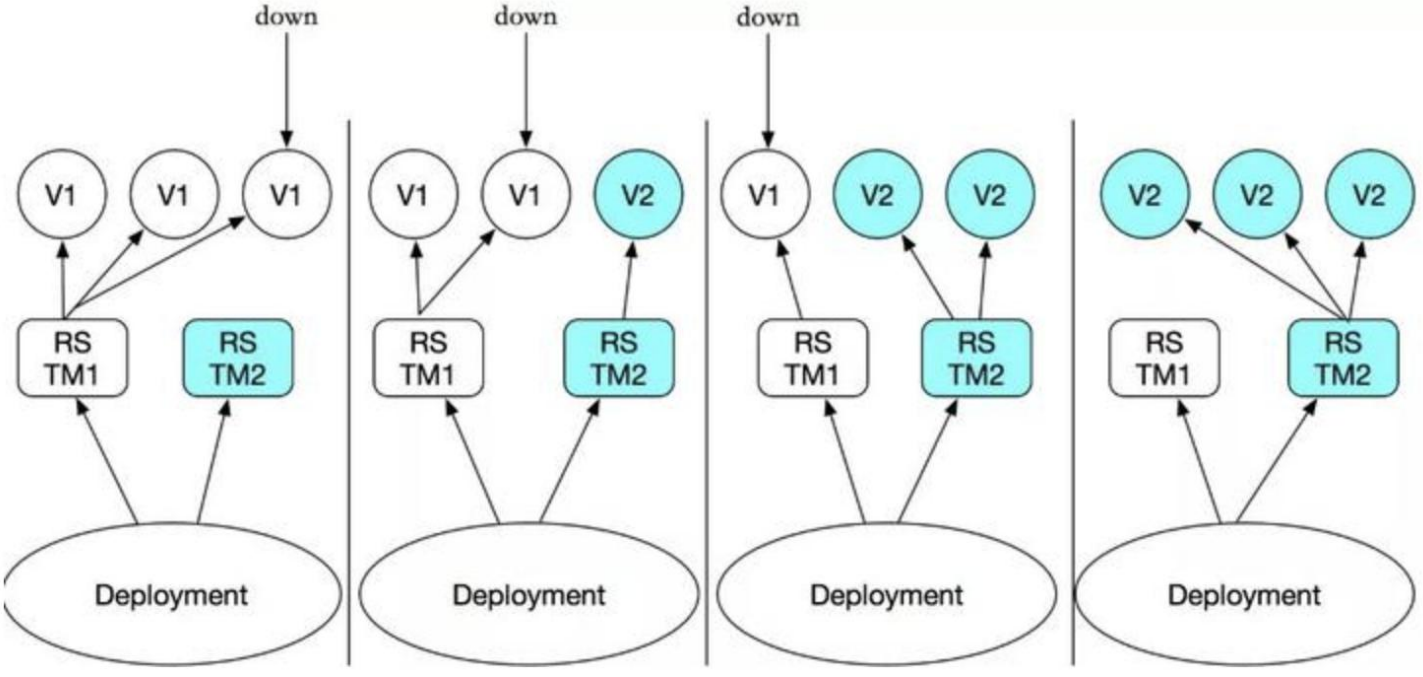
- 容器按照持续运行时间，可以分为服务类容器和工作类容器。
- 服务类容器通常持续提供服务，需要一直运行，比如HTTP Server、Daemon等。工作类容器则是一次性任务，比如批处理程序，完成后容器就退出。
- Kubernetes的Deployment、ReplicaSet和DaemonSet都用于管理服务类容器；
- 对于工作类容器，我们使用Job。

Service

- 我们不应该期望Pod是健壮的，而是要假设Pod中的容器很可能因为各种原因发生故障而死掉。
- Deployment等Controller会通过动态创建和销毁Pod来保证应用整体的健壮性。换句话说，Pod是脆弱的，但应用是健壮的。
- Service提供了固定的ip和端口，并且里面包含一组pod，即使Pod的ip发生变化，但是面对客户端的是Service的固定ip和端口。

Rolling Update

Rolling Update



滚动更新是一次只更新一小部分副本，成功后再更新更多的副本，最终完成所有副本的更新。滚动更新的最大好处是零停机，整个更新过程始终有副本在运行，从而保证了业务的连续性。

结束