Taint&Toleration：

1. 在不同的机房
2. 在不同的城市
3. 有着不一样配置
   1. GPU服务器
   2. 纯固态硬盘的服务器
4. NodeSelect：
   1. Gpu-server：true
   2. Ssd-server：true
   3. Normal-server：true

污点和容忍的理念：

Taint在一类服务器上打上污点，让不能容忍这个污点的Pod不能部署在打了污点的服务器上。

Master节点不应该部署系统Pod之外的任何Pod。

每个节点可以打很多个污点。

GPU： gpu-server: true

给一个节点打一个污点：

[root@k8s-master01 ~]# kubectl taint node k8s-master01 master-test=test:NoSchedule

NoSchedule：禁止调度

NoExecute：如果不符合这个污点，会立马被驱逐

PreferNoSchedule: 尽量避免将Pod调度到指定的节点上。

tolerations:

- effect: NoSchedule

key: master-test

operator: Equal

value: test

Node节点有多个Taint，每个Taint都需要容忍才能部署上去。

tolerations:

- effect: NoSchedule

key: master-test

operator: Exists

tolerations:

- operator: Exists

tolerations:

- operator: Exists

key: master-test

node.kubernetes.io/not-ready: 节点没有准备好，Ready不为true

InitContainer：

初始化容器：在我应用容器启动之前做的一些舒适化操作。

postStart：在容器启动之前做一些操作。不能保证在你的container的EntryPoint

Affinity：亲和力。

* NodeAffinity：节点亲和力，
  + - RequiredDuringSchedulingIgnoredDuringExecution：硬亲和力，即支持必须部署在指定的节点上，也支持必须不部署在指定的节点上。

a=b

* + - PreferredDuringSchedulingIgnoredDuringExecution：软亲和力，尽量部署在满足条件的节点上，或者是尽量不要部署在被匹配的节点。
* PodAffinity：Pod亲和力
  + A应用B应用C应用，将A应用根据某种策略尽量或者部署在一块。Label
    - A：app=a B：app=b
    - RequiredDuringSchedulingIgnoredDuringExecution：
      * 将A应用和B应用部署在一块
    - PreferredDuringSchedulingIgnoredDuringExecution：
      * 尽量将A应用和B应用部署在一块
* PodAntiAffinity：Pod反亲和力
  + A应用B应用C应用，将A应用根据某种策略尽量或不部署在一块。Label
    - RequiredDuringSchedulingIgnoredDuringExecution：
      * 不要将A应用与与之匹配的应用部署在一块
    - PreferredDuringSchedulingIgnoredDuringExecution
      * 尽量。。。

NodeAffinity：

apiVersion: apps/v1

kind: Deployment

metadata:

labels:

app: demo-nginx

name: demo-nginx

spec:

progressDeadlineSeconds: 600

replicas: 2

revisionHistoryLimit: 10

selector:

matchLabels:

app: demo-nginx

strategy:

rollingUpdate:

maxSurge: 25%

maxUnavailable: 1

type: RollingUpdate

template:

metadata:

creationTimestamp: null

labels:

app: demo-nginx

spec:

affinity:

nodeAffinity:

requiredDuringSchedulingIgnoredDuringExecution:

nodeSelectorTerms:

- matchExpressions:

- key: kubernetes.io/e2e-az-name

operator: In

values:

- e2e-az1

- e2e-az2

preferredDuringSchedulingIgnoredDuringExecution:

- weight: 1

preference:

matchExpressions:

- key: another-node-label-key

operator: In

values:

- another-node-label-value

containers:

- command:

- sh

- -c

- sleep 36000000000

image: nginx

imagePullPolicy: IfNotPresent

name: nginx2

resources: {}

terminationMessagePath: /dev/termination-log

terminationMessagePolicy: File

volumeMounts:

- mountPath: /mnt

name: cache-volume

- mountPath: /tmp/nfs

name: nfs-test

- command:

- sh

- -c

- sleep 36000000000

image: nginx

imagePullPolicy: IfNotPresent

name: nginx

ports:

- containerPort: 80

name: web

protocol: TCP

resources:

limits:

cpu: 100m

memory: 270Mi

requests:

cpu: 100m

memory: 70Mi

terminationMessagePath: /dev/termination-log

terminationMessagePolicy: File

volumeMounts:

- mountPath: /etc/nginx/nginx.conf

name: config-volume

subPath: etc/nginx/nginx.conf

- mountPath: /mnt/

name: config-volume-non-subpath

- mountPath: /tmp/1

name: test-hostpath

- mountPath: /tmp/2

name: cache-volume

dnsPolicy: ClusterFirst

initContainers:

- command:

- sh

- -c

- echo "InitContainer" >> /tmp/nfs/init

image: nginx

imagePullPolicy: IfNotPresent

name: init1

resources: {}

terminationMessagePath: /dev/termination-log

terminationMessagePolicy: File

volumeMounts:

- mountPath: /tmp/nfs

name: nfs-test

restartPolicy: Always

schedulerName: default-scheduler

securityContext: {}

shareProcessNamespace: true

terminationGracePeriodSeconds: 30

tolerations:

- effect: NoSchedule

key: master-test

operator: Equal

value: test

- effect: NoExecute

key: master-test

operator: Equal

tolerationSeconds: 60

value: test

volumes:

- hostPath:

path: /etc/hosts

type: File

name: test-hostpath

- configMap:

defaultMode: 420

items:

- key: nginx.conf

path: etc/nginx/nginx.conf

name: nginx-conf

name: config-volume

- configMap:

defaultMode: 420

name: nginx-conf

name: config-volume-non-subpath

- emptyDir:

medium: Memory

name: cache-volume

- name: nfs-test

nfs:

path: /data/k8s-data/testDir

server: 192.168.1.22

In：部署在满足多个条件的节点上

NotIn：不要部署在满足这些条件的节点上

Exists：部署在具有某个存在key为指定的值的Node节点上

DoesNotExist：和Exists相反

Gt： 大于指定的条件 条件为number，不能为字符串

Lt：小于指定的条件

PodAffinity：

apiVersion: apps/v1

kind: Deployment

metadata:

labels:

app: demo-nginx

name: demo-nginx

namespace: default

spec:

progressDeadlineSeconds: 600

replicas: 2

revisionHistoryLimit: 10

selector:

matchLabels:

app: demo-nginx

strategy:

rollingUpdate:

maxSurge: 25%

maxUnavailable: 1

type: RollingUpdate

template:

metadata:

creationTimestamp: null

labels:

app: demo-nginx

spec:

# 把demo-nginx和kube-systemnamespace下的符合label为k8s-app= calico-kube-controllers的Pod部署在同一个节点（拓扑域）上

affinity:

podAffinity:

requiredDuringSchedulingIgnoredDuringExecution:

- labelSelector:

matchExpressions:

- key: k8s-app

operator: In

values:

- calico-kube-controllers

# 如果写了namespaces的字段，但是留空，他是匹配所有namespace下的指定label的Pod，如果写了namespace并且指定了值，就是匹配指定namespace下的指定label的Pod。如果没有写namespace，匹配当前namespace

# namespaces:

namespaces:

- kube-system

topologyKey: kubernetes.io/hostname

containers:

- command:

- sh

- -c

- sleep 36000000000

image: nginx

imagePullPolicy: IfNotPresent

name: nginx2

resources: {}

terminationMessagePath: /dev/termination-log

terminationMessagePolicy: File

volumeMounts:

- mountPath: /mnt

name: cache-volume

- mountPath: /tmp/nfs

name: nfs-test

- command:

- sh

- -c

- sleep 36000000000

image: nginx

imagePullPolicy: IfNotPresent

name: nginx

ports:

- containerPort: 80

name: web

protocol: TCP

resources:

limits:

cpu: 100m

memory: 270Mi

requests:

cpu: 100m

memory: 70Mi

terminationMessagePath: /dev/termination-log

terminationMessagePolicy: File

volumeMounts:

- mountPath: /etc/nginx/nginx.conf

name: config-volume

subPath: etc/nginx/nginx.conf

- mountPath: /mnt/

name: config-volume-non-subpath

- mountPath: /tmp/1

name: test-hostpath

- mountPath: /tmp/2

name: cache-volume

dnsPolicy: ClusterFirst

restartPolicy: Always

schedulerName: default-scheduler

securityContext: {}

shareProcessNamespace: true

terminationGracePeriodSeconds: 30

tolerations:

- effect: NoSchedule

key: master-test

operator: Equal

value: test

- effect: NoExecute

key: master-test

operator: Equal

tolerationSeconds: 60

value: test

volumes:

- hostPath:

path: /etc/hosts

type: File

name: test-hostpath

- configMap:

defaultMode: 420

items:

- key: nginx.conf

path: etc/nginx/nginx.conf

name: nginx-conf

name: config-volume

- configMap:

defaultMode: 420

name: nginx-conf

name: config-volume-non-subpath

- emptyDir:

medium: Memory

name: cache-volume

- name: nfs-test

nfs:

path: /data/k8s-data/testDir

server: 192.168.1.22

topologyKey: kubernetes.io/hostname

topologyKey：拓扑域，首先说明一点不同的key不同的value是属于不同的拓扑域。

affinity:

podAffinity:

requiredDuringSchedulingIgnoredDuringExecution:

- labelSelector:

matchExpressions:

- key: k8s-app

operator: In

values:

- calico-kube-controllers

# 如果写了namespaces的字段，但是留空，他是匹配所有namespace下的指定label的Pod，如果写了namespace并且指定了值，就是匹配指定namespace下的指定label的Pod。如果没有写namespace，匹配当前namespace

# namespaces:

namespaces:

- kube-system

topologyKey: kubernetes.io/hostname

kube-system🡪k8s-app=calico-kube-controllers

affinity:

podAntiAffinity:

preferredDuringSchedulingIgnoredDuringExecution:

- podAffinityTerm:

labelSelector:

matchExpressions:

- key: app

operator: In

values:

- demo-nginx

topologyKey: jigui

weight: 1