## Volumes

Container（容器）中的磁盘文件是短暂的，当容器崩溃时，kubelet会重新启动容器，但最初的文件将丢失，Container会以最干净的状态启动。另外，当一个Pod运行多个Container时，各个容器可能需要共享一些文件。Kubernetes Volume可以解决这两个问题。

一些需要持久化数据的程序才会用到Volumes，或者一些需要共享数据的容器需要volumes。

Redis-Cluster：nodes.conf

日志收集的需求：需要在应用程序的容器里面加一个sidecar，这个容器是一个收集日志的容器，比如filebeat，它通过volumes共享应用程序的日志文件目录。

Volumes：官方文档<https://kubernetes.io/docs/concepts/storage/volumes/>

1. 背景

Docker也有卷的概念，但是在Docker中卷只是磁盘上或另一个Container中的目录，其生命周期不受管理。虽然目前Docker已经提供了卷驱动程序，但是功能非常有限，例如从Docker 1.7版本开始，每个Container只允许一个卷驱动程序，并且无法将参数传递给卷。

另一方面，Kubernetes卷具有明确的生命周期，与使用它的Pod相同。因此，在Kubernetes中的卷可以比Pod中运行的任何Container都长，并且可以在Container重启或者销毁之后保留数据。Kubernetes支持多种类型的卷，Pod可以同时使用任意数量的卷。

从本质上讲，卷只是一个目录，可能包含一些数据，Pod中的容器可以访问它。要使用卷Pod需要通过.spec.volumes字段指定为Pod提供的卷，以及使用.spec.containers.volumeMounts 字段指定卷挂载的目录。从容器中的进程可以看到由Docker镜像和卷组成的文件系统视图，卷无法挂载其他卷或具有到其他卷的硬链接，Pod中的每个Container必须独立指定每个卷的挂载位置。

### emptyDir

和上述volume不同的是，如果删除Pod，emptyDir卷中的数据也将被删除，一般emptyDir卷用于Pod中的不同Container共享数据。它可以被挂载到相同或不同的路径上。

默认情况下，emptyDir卷支持节点上的任何介质，可能是SSD、磁盘或网络存储，具体取决于自身的环境。可以将emptyDir.medium字段设置为Memory，让Kubernetes使用tmpfs（内存支持的文件系统），虽然tmpfs非常快，但是tmpfs在节点重启时，数据同样会被清除，并且设置的大小会被计入到Container的内存限制当中。

使用emptyDir卷的示例，直接指定emptyDir为{}即可：

# cat nginx-deploy.yaml

apiVersion: apps/v1

kind: Deployment

metadata:

annotations:

deployment.kubernetes.io/revision: "1"

creationTimestamp: "2020-09-19T02:41:11Z"

generation: 1

labels:

app: nginx

name: nginx

namespace: default

spec:

progressDeadlineSeconds: 600

replicas: 2

revisionHistoryLimit: 10

selector:

matchLabels:

app: nginx

strategy:

rollingUpdate:

maxSurge: 25%

maxUnavailable: 25%

type: RollingUpdate

template:

metadata:

creationTimestamp: null

labels:

app: nginx

spec:

containers:

- image: nginx:1.15.2

imagePullPolicy: IfNotPresent

name: nginx

resources: {}

terminationMessagePath: /dev/termination-log

terminationMessagePolicy: File

terminationMessagePath: /dev/termination-log

terminationMessagePolicy: File

volumeMounts:

- mountPath: /opt

name: share-volume

- image: nginx:1.15.2

imagePullPolicy: IfNotPresent

name: nginx2

command:

- sh

- -c

- sleep 3600

resources: {}

terminationMessagePath: /dev/termination-log

terminationMessagePolicy: File

terminationMessagePath: /dev/termination-log

terminationMessagePolicy: File

volumeMounts:

- mountPath: /mnt

name: share-volume

dnsPolicy: ClusterFirst

restartPolicy: Always

schedulerName: default-scheduler

securityContext: {}

terminationGracePeriodSeconds: 30

volumes:

- name: share-volume

emptyDir: {}

#medium: Memory

### hostPath

hostPath卷可将节点上的文件或目录挂载到Pod上，用于Pod自定义日志输出或访问Docker内部的容器等。

使用hostPath卷的示例。将主机的/data目录挂载到Pod的/test-pd目录：

# cat nginx-deploy.yaml

apiVersion: apps/v1

kind: Deployment

metadata:

annotations:

deployment.kubernetes.io/revision: "1"

creationTimestamp: "2020-09-19T02:41:11Z"

generation: 1

labels:

app: nginx

name: nginx

namespace: default

spec:

progressDeadlineSeconds: 600

replicas: 2

revisionHistoryLimit: 10

selector:

matchLabels:

app: nginx

strategy:

rollingUpdate:

maxSurge: 25%

maxUnavailable: 25%

type: RollingUpdate

template:

metadata:

creationTimestamp: null

labels:

app: nginx

spec:

containers:

- image: nginx:1.15.2

imagePullPolicy: IfNotPresent

name: nginx

resources: {}

terminationMessagePath: /dev/termination-log

terminationMessagePolicy: File

terminationMessagePath: /dev/termination-log

terminationMessagePolicy: File

volumeMounts:

- mountPath: /opt

name: share-volume

- mountPath: /etc/timezone

name: timezone

- image: nginx:1.15.2

imagePullPolicy: IfNotPresent

name: nginx2

command:

- sh

- -c

- sleep 3600

resources: {}

terminationMessagePath: /dev/termination-log

terminationMessagePolicy: File

terminationMessagePath: /dev/termination-log

terminationMessagePolicy: File

volumeMounts:

- mountPath: /mnt

name: share-volume

dnsPolicy: ClusterFirst

restartPolicy: Always

schedulerName: default-scheduler

securityContext: {}

terminationGracePeriodSeconds: 30

volumes:

- name: share-volume

emptyDir: {}

#medium: Memory

- name: timezone

hostPath:

path: /etc/timezone

type: **File**

hostPath卷常用的type（类型）如下：

* type为空字符串：默认选项，意味着挂载hostPath卷之前不会执行任何检查。
* DirectoryOrCreate：如果给定的path不存在任何东西，那么将根据需要创建一个权限为0755的空目录，和Kubelet具有相同的组和权限。
* Directory：目录必须存在于给定的路径下。
* FileOrCreate：如果给定的路径不存储任何内容，则会根据需要创建一个空文件，权限设置为0644，和Kubelet具有相同的组和所有权。
* File：文件，必须存在于给定路径中。
* Socket：UNIX套接字，必须存在于给定路径中。
* CharDevice：字符设备，必须存在于给定路径中。
* BlockDevice：块设备，必须存在于给定路径中。

### NFS

Exporter配置

/data/nfs/ 192.168.0.0/24(rw,sync,no\_subtree\_check,no\_root\_squash)

# cat nginx-deploy.yaml

apiVersion: apps/v1

kind: Deployment

metadata:

annotations:

deployment.kubernetes.io/revision: "1"

creationTimestamp: "2020-09-19T02:41:11Z"

generation: 1

labels:

app: nginx

name: nginx

namespace: default

spec:

progressDeadlineSeconds: 600

replicas: 2

revisionHistoryLimit: 10

selector:

matchLabels:

app: nginx

strategy:

rollingUpdate:

maxSurge: 25%

maxUnavailable: 25%

type: RollingUpdate

template:

metadata:

creationTimestamp: null

labels:

app: nginx

spec:

containers:

- image: nginx:1.15.2

imagePullPolicy: IfNotPresent

name: nginx

resources: {}

terminationMessagePath: /dev/termination-log

terminationMessagePolicy: File

terminationMessagePath: /dev/termination-log

terminationMessagePolicy: File

volumeMounts:

- mountPath: /opt

name: share-volume

- mountPath: /etc/timezone

name: timezone

- image: nginx:1.15.2

imagePullPolicy: IfNotPresent

name: nginx2

command:

- sh

- -c

- sleep 3600

resources: {}

terminationMessagePath: /dev/termination-log

terminationMessagePolicy: File

terminationMessagePath: /dev/termination-log

terminationMessagePolicy: File

volumeMounts:

- mountPath: /mnt

name: share-volume

- mountPath: /opt

name: nfs-volume

dnsPolicy: ClusterFirst

restartPolicy: Always

schedulerName: default-scheduler

securityContext: {}

terminationGracePeriodSeconds: 30

volumes:

- name: share-volume

emptyDir: {}

#medium: Memory

- name: timezone

hostPath:

path: /etc/timezone

type: File

- name: nfs-volume

nfs:

server: 192.168.0.204

path: /data/nfs/test-dp