StatefulSet管理与使用

前言

- 本章节通过有状态服务和无状态服务的特性对比，引入了Kubernetes对有状态服务的解决方案——StatefulSet。重点介绍了如何通过StatefulSet构建有状态应用。

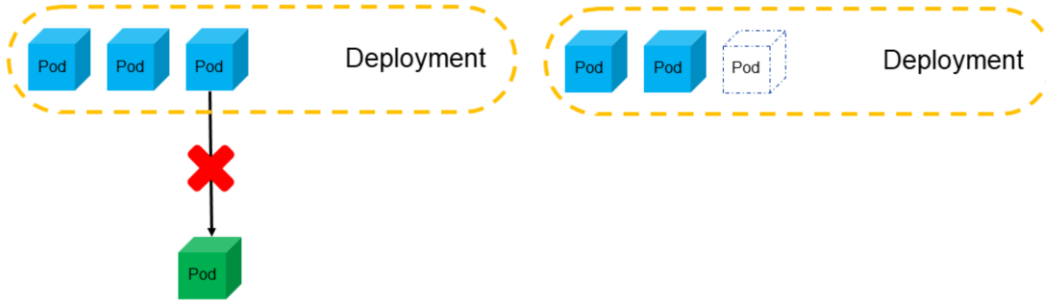
目标

- 学完本课程后，您将能够：
  - 描述什么是有状态应用
  - 描述StatefulSet的特征
  - 使用StatefulSet构建有状态应用

目录

Deployment的特征

- 回顾Deployment，可以发现它有一些很有趣的特征，比如，所有Pod地位都是平等的，当一个Pod因故障被替换后，新的容器与其余所有Pod依然相同，因此无论请求发送到哪个Pod，返回的结果都是一致的。又或者Pod被删除后，里面的数据也随之消失。这种特性一般称其为“无状态”。



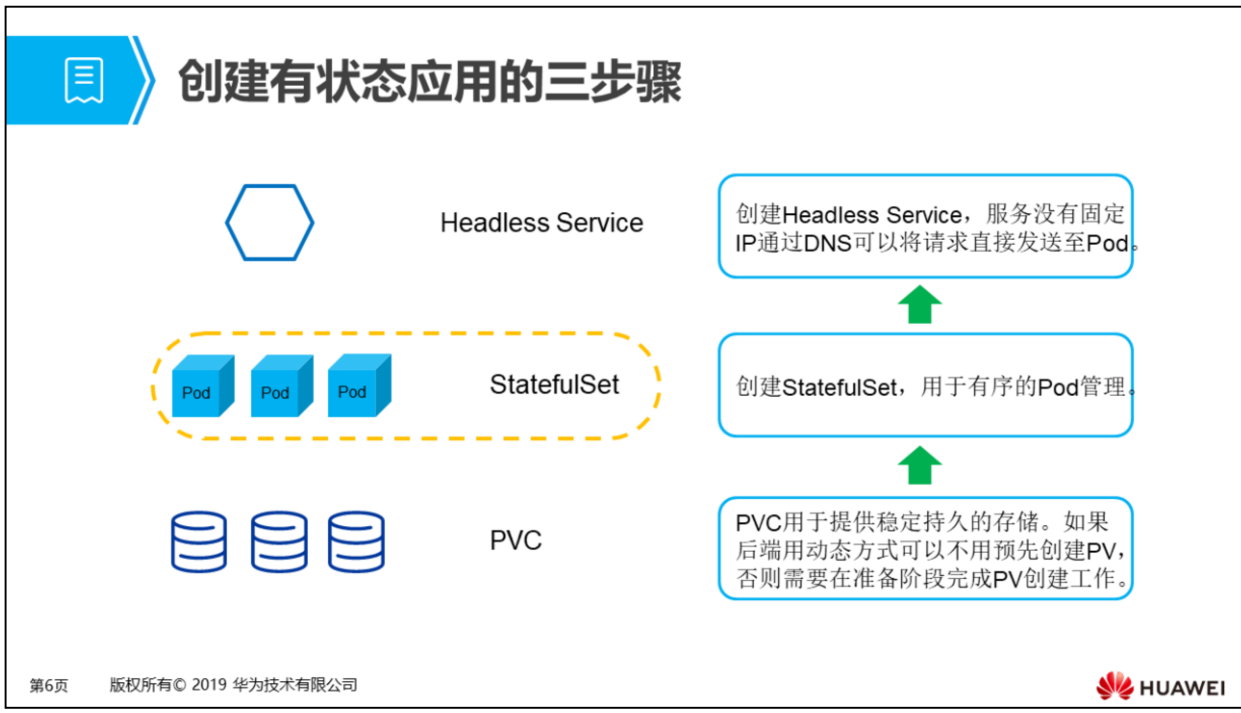
HUAWEI

- 无状态：stateless

“有状态”的应用

- 有些情况下希望运行一些和上述不同的应用，包括使用持久化的存储，稳定的网络标志，Pod与Pod之间并不是完全平等（即使它们用同一个镜像创建）。这类服务通常称为“有状态”的服务，它的实现不再依靠ReplicaSet，而是用StatefulSet，它具备以下特性：
  - 稳定的持久化存储，Pod重新调度后访问相同的持久化数据，使用PVC来实现。
  - 稳定的网络标识，Pod重新调度后PodName和HostName不变，基于Headless Service实现。
  - Pod都有一个“序号”，可以有序的进行扩展，部署等操作。

- 有状态：stateful



- 有状态应用一般需要绑定固定的存储，用户访问时会被转发流量到特定的Pod。

创建PV和PVC

- 预先创建好一批PV

```
NAME    CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS      CLAIM   STORAGECLASS   AGE
mypv1   1Gi        RWO            Recycle          Available           my-sc          2m58s
mypv2   1Gi        RWO            Recycle          Available           my-sc          17s
mypv3   1Gi        RWO            Recycle          Available           my-sc          15s
```

- 在yaml文件中定义PVC的模板，使系统在创建statefulset时自动创建PVC。

```
volumeClaimTemplates:
- metadata:
    name: stor
  spec:
    accessModes: [ReadWriteOnce]
    storageClassName: my-sc
    resources:
      requests:
        storage: 1Gi
```


- 创建PVC的方式和前面存储章节介绍的有所不同，具体可以参考实验手册。

创建StatefulSet

- 创建StatefulSet时需要注意，serviceName这一项的值必须要和后面创建的headless Service的名称一致。
- 另外matchLabels参数和pod的标签一致，这与deployment相同。
- 上一步关于volumeClaimTemplates的配置也需要添加进StatefulSet配置yaml中。

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: web
spec:
  selector:
    matchLabels:
      app: nginx
  serviceName: nginx
  replicas: 3
  template:
    metadata:
      labels:
        app: nginx
    spec:
      terminationGracePeriodSeconds: 10
      containers:
      - name: nginx
        image: nginx
        ports:
        - containerPort: 80
          name: web
        volumeMounts:
        - name: stor
          mountPath: /usr/share/nginx/html
```


HUAWEI

- Headless service中标签选择器要与Pod的标签一致，注意不是和statefulSet一致。

创建结果

- 创建完成后，可以看到Pod名称按序号排序。如果查看日志，可以发现Pod的创建顺序是第一个创建完成之后再创建第二个，依次完成。

```
[root@k8s-master stateful]# kubectl get pod
NAME    READY   STATUS    RESTARTS   AGE
web-0   1/1     Running   0          5m28s
web-1   1/1     Running   0          5m21s
web-2   1/1     Running   0          4m56s
```

- 创建了三个PVC，名称同样以序号排序，依次和PV做绑定。

```
[root@k8s-master stateful]# kubectl get pvc
NAME          STATUS   VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS   AGE
stor-web-0    Bound    mypv1    1Gi        RWO            my-sc          65s
stor-web-1    Bound    mypv2    1Gi        RWO            my-sc          58s
stor-web-2    Bound    mypv3    1Gi        RWO            my-sc          33s
```


- 如果PVC创建失败则不会创建Pod

创建Headless服务

- 创建headless服务，注意以下几点：
  - 服务名称和statefulset中的定义一致。
  - 选择器要指向正确的Pod标签。
  - 指定clusterIP：None

```
apiVersion: v1
kind: Service
metadata:
  name: nginx
  labels:
    app: nginx
spec:
  ports:
  - port: 80
    name: web
  clusterIP: None
  selector:
    app: nginx
```


- Headless服务的yaml编写可以参考service章节

使用有状态的服务

- 使用nslookup查看DNS记录，可以看到对该服务的访问直接指向Pod。

```
Name:      nginx
Address 1: 10.244.1.99 web-2.nginx.default.svc.cluster.local
Address 2: 10.244.2.120 web-0.nginx.default.svc.cluster.local
Address 3: 10.244.0.25 web-1.nginx.default.svc.cluster.local
```

- 通过对稳定的名称访问也始终可以访问到固定的Pod。

```
Name:      web-0.nginx
Address 1: 10.244.2.120 web-0.nginx.default.svc.cluster.local
```


- Nslookup命令需要在pod中运行。

StatefulSet的故障处理

- 与无状态服务不同，有状态的StatefulSet中一个Pod出现故障之后，可以看到
  - 虽然Pod的IP地址变化了，但通过不变的域名，依然可以访问到重建后的Pod。

```
Name:      web-1.nginx
Address 1: 10.244.0.26 web-1.nginx.default.svc.cluster.local
```

  - 即使经过了故障和重建，Pod中保持在持久化存储中的数据依然存在。

```
# cd /usr/share/nginx/html
# ls
newfile
```


- StatefulSet经历故障后，只需要自动重建Pod，状态不会丢失。

扩缩容和升级

- 当缩容StatefulSet时，可以看到Pod停止的顺序为从序号最高的开始降序终止，并且只有在前一个pod被完全终止后，下一个pod才开始终止。升级时，也是以相同顺序处理。

```
NAME      READY   STATUS        RESTARTS   AGE
web-0     1/1     Running       0          2d16h
web-1     1/1     Running       0          59m
web-2     1/1     Terminating   0          2d20h
web-2     0/1     Terminating   0          2d20h
web-2     0/1     Terminating   0          2d20h
web-2     0/1     Terminating   0          2d20h
web-1     1/1     Terminating   0          60m
web-1     0/1     Terminating   0          60m
web-1     0/1     Terminating   0          60m
web-1     0/1     Terminating   0          60m
web-0     1/1     Terminating   0          2d16h
web-0     0/1     Terminating   0          2d16h
web-0     0/1     Terminating   0          2d16h
web-0     0/1     Terminating   0          2d16h
```


- 删除stateful时将会并行删除pod，如果需要有序优雅地删除pod，可以先使用缩容为0，再删除statefulset的方式。

Pod管理策略

- 对于某些分布式系统，StatefulSet的顺序性保证是不必要的，或者是不应该出现的。为了解决这个问题，在Kubernetes 1.7中引入了podManagementPolicy。
  - OrderedReady Pod管理策略
    StatefulSets的默认选项，保证顺序性。
  - Parallel Pod管理策略：
    并行启动、终止、升级、弹性伸缩Pod。
    在变更一个Pod时不必等前一个Pod完成。

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: web
spec:
  selector:
    matchLabels:
      app: nginx
  serviceName: nginx
  podManagementPolicy: Parallel
  replicas: 3
  ……
```


- 即使采用了parallel的Pod管理策略，statefulset和deployment依然有很大区别。

实验&实训任务

- 实验任务
  - 请按照实验手册2.12章节完成StatefulSet相关实验，包括：
    - 创建StatefulSet
    - 使用StatefulSet
- 实训任务
  - 请灵活使用本章节课程及实验手册中学到的知识，按照实验手册2.12.3章节完成StatefulSet相关实训任务。

本章总结

- 本章节学习后，您是否了解了有状态应用与无状态应用的区别？是否能够一步步地创建有状态应用呢？


谢 谢
www.huawei.com