DaemonSet与Job

前言

- 本章节介绍了DaemonSet与Job，这是和Deployment有所区别的Pod控制器，本章节我们将掌握这几种对象的特性和使用方式。

目标

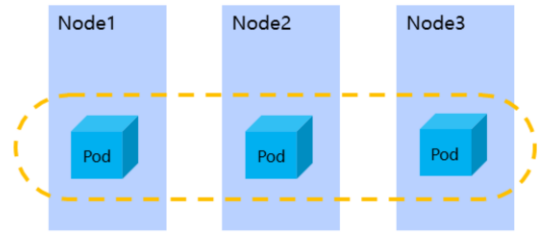
- 学完本课程后，您将能够：
  - 描述DaemonSet特性和使用方式
  - 区分Job和CronJob
  - 使用Job和CronJob

目录

1. **DaemonSet**
2. Job
3. CronJob

Kube-proxy的特殊性

- 在Kubernetes组件章节中，介绍了kube-proxy组件，它被封装在Pod中，并且时刻运行在每个Node节点中。
- ReplicaSet侧重保证Pod数量的恒定，那该如何实现kube-proxy类应用的调度?




```
NAME                READY   STATUS    RESTARTS   AGE    IP               NODE
kube-proxy-5q6mg    1/1     Running   0          125m   192.168.137.21   k8s-master
kube-proxy-8dt55    1/1     Running   0          125m   192.168.137.23   k8s-node2
kube-proxy-m4dx6    1/1     Running   0          125m   192.168.137.22   k8s-node1
```


HUAWEI

- Daemonset保证在所有承担业务的节点上运行Pod。

DaemonSet的特性

- DaemonSet部署的副本Pod会分布在各个Node上。当有Node加入集群时，也会为他们新增一个Pod。当有Node从集群移除时，这些Pod也会被回收。删除DaemonSet将会删除它创建的所有Pod。
- DaemonSet典型场景：
  - 在集群的每个节点上运行存储Daemon，如glusterd，ceph。
  - 在每个节点上运行日志收集Daemon，如flunentd或logstash。
  - 在每个节点上运行监控Daemon，如Prometheus Node Exporter。

创建DaemonSet

```
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: nginx-daemonset
spec:
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
       app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.7.9
        ports:
        - containerPort: 80
```

- 创建DaemonSet的yaml文件和创建deployment使用的类似。
- 类型（kind）选择DaemonSet。
- DaemonSet的yaml文件不需要副本数量项。
- 默认情况下，DaemonSet会在所有Node上创建Pod。

- 一般nginx不需要用daemonset创建，此处仅仅是为了演示。

DaemonSet查看

- 创建完成后，可以在Node1和Node2上分别看到一个pod。

```
[root@k8s-master runfile]# kubectl get pod -o wide
NAME                    READY   STATUS    RESTARTS   AGE   IP            NODE
nginx-daemonset-ftvj8   1/1     Running   0          84m   10.244.2.13   k8s-node2
nginx-daemonset-r5xfl   1/1     Running   0          84m   10.244.1.10   k8s-node1
```

- 如将其中一个pod强制删除，daemonset会自动启动一个新的pod。

```
nginx-daemonset-ftvj8   1/1     Running   0          88m   10.244.2.13   k8s-node2
nginx-daemonset-vmgkw   1/1     Running   0          7s    10.244.1.12   k8s-node1
```

- 当Kubernetes集群出现节点故障时，daemonset中的pod数量会减少，不会像deployment一样将受到影响的pod在其他节点启动。

```
[root@k8s-master runfile]# kubectl get daemonset
NAME              DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR   AGE
nginx-daemonset   1         1         1       1            1           <none>          133m
```

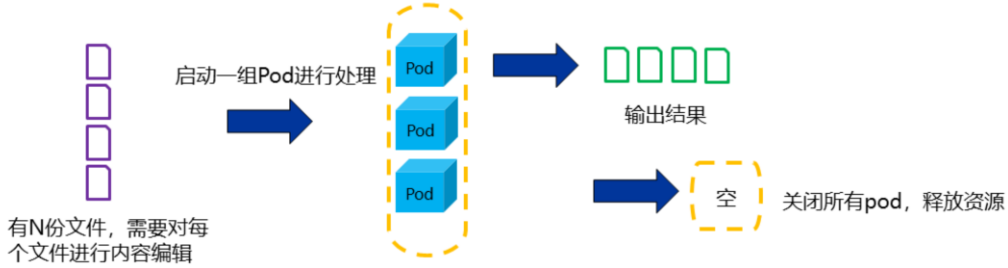

- 在故障节点恢复后，daemonset副本数量是多少?

目录

1. DaemonSet
2. **Job**
3. CronJob

一次性任务场景

- 通过Deployment我们可以部署常驻型应用，它可以保证pod数量保证应用的实时可用，也可以通过灵活的扩缩容让业务处于最佳状态。
- 通过DaemonSet我们可以部署守护进程，它使得每个node上运行着一个固定pod。
- 应该使用哪种方式解决如下问题?



HUAWEI

- Job页是一种副本控制器。

运行一个Job

- 相对于Deployment和DaemonSet通常提供持续的服务，Job执行一次性任务。

```
apiVersion: batch/v1
kind: Job
metadata:
  name: pi
spec:
  template:
    spec:
      containers:
      - name: pi
        image: perl
        command: ["perl",  "-Mbignum=bpi", "-wle", "print bpi(2000)"]
      restartPolicy: Never
  backoffLimit: 4
```

- 类型（kind）选择Job。
- restartPolicy只能选择Never或OnFailure。
- backoffLimit:参数指定job失败后进行重试的次数。

- 在deployment中，restartPolicy我们还可以选择always

查看Job

- 运行job后，查看状态

```
[root@k8s-master runfile]# kubectl get job
NAME   COMPLETIONS   DURATION   AGE
pi     0/1           5s         5s
```

- 过一段时间后继续查看状态，completions状态从“0/1”变为“1/1”，表示任务已经完成。

```
[root@k8s-master runfile]# kubectl get job
NAME   COMPLETIONS   DURATION   AGE
pi     1/1           18s        56s
```

- 查看job使用的pod状态，发现pod已经运行完成并且关闭。

```
[root@k8s-master runfile]# kubectl get pod
NAME                    READY   STATUS      RESTARTS   AGE
nginx-daemonset-fjz7r   1/1     Running     0          4h23m
pi-nx9m9                0/1     Completed   0          117s
```

- 使用kubectl logs命令可以查看该Job运行结果。

- 使用logs命令可以查看job运行结果

目录

1. DaemonSet
2. Job
3. **CronJob**

运行CronJob

```
apiVersion: batch/v1beta1
kind: CronJob
metadata:
  name: hello
spec:
  schedule: "*/1 * * * *"
  jobTemplate:
    spec:
      template:
        spec:
          containers:
          - name: hello
            image: busybox
            args:
            - /bin/sh
            - -c
            - date; echo Hello from the
Kubernetes cluster
          restartPolicy: OnFailure
```

- 在日常应用中，有一种常见场景是需要Job在指定时间或周期运行，这种类型我们称其为Cron Job，主要管理基于时间的Job。
  - 在给定时间点只运行一次
  - 在给定时间点周期性地运行
- 使用配置文件创建Cron Job。

- Cron Job不是Job的一种，属于不同Kind

Schedule参数配置

- Schedule的格式和linux中crontab命令类似。

```
┌──────────── minute (0 - 59)
│ ┌────────── hour (0 - 23)
│ │ ┌──────── day of the month (1 - 31)
│ │ │ ┌────── month (1 - 12)
│ │ │ │ ┌──── day of the week (0 - 6) (Sunday to Saturday;
│ │ │ │ │                             7 is also Sunday on some systems)
│ │ │ │ │
* * * * *
```

- 如果需要在每个小时的第15分钟执行任务

```
15 * * * *
```

- 如果需要每15分钟执行一次任务

```
*/15 * * * *
```


- 具体语法参考crontab命令的语法。

实验&实训任务

- 实验任务
  - 请按照实验手册中2.7章节完成DaemonSet和Job相关实验，包括：
    - 使用DaemonSet
    - 使用Job
    - 使用Cron Job
- 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，按照实验手册中2.7.4章节完成DaemonSet和Job的实训任务。

本章总结

- 本章节介绍了如下几种Pod控制器的概念和使用方式：
  - DaemonSet
  - Job
  - CronJob


谢 谢
www.huawei.com