Kubernetes Replicaset 控制器

前言：

课程名称：Kubernetes Replicaset 控制器

实验环境：

本章节 Kubernetes 集群环境如下：

角色	IP	主机名	组件	硬件
控制节点	192.168.128.11	k8s-master01	apiserver controller-manager scheduler etcd containerd	CPU：4vCPU 硬盘：100G 内存：4GB 开启虚拟化
工作节点	192.168.128.21	k8s-node01	kubelet kube-proxy containerd calico coredns	CPU：6vCPU 硬盘：100G 内存：8GB 开启虚拟化
工作节点	192.168.128.22	k8s-node02	kubelet kube-proxy containerd calico coredns	CPU：6vCPU 硬盘：100G 内存：8GB 开启虚拟化

张岩峰老师微信，加我微信，邀请你加入 VIP 交流答疑群：

微信号：ZhangYanFeng0429

二维码：

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。

1、前言

前面我们学习了 Pod，那我们在定义 pod 资源时，可以直接创建一个 kind：Pod 类型的自主式 pod，但是这存在一个问题，假如 pod 被删除了，那这个 pod 就不能自我恢复，就会彻底被删除，在生产环境这种情况是非常危险的，所以今天就给大家讲解下 pod 的控制器，所谓控制器就是能够管理 pod，监测 pod 运行状况，当 pod 发生故障，可以自动恢复 pod。也就是说能够代我们去管理 pod 中间层，并帮助我们确保每一个 pod 资源始终处于我们所定义或者我们所期望的目标状态，一旦 pod 资源出现故障，那么控制器会尝试重启 pod 或者里面的容器，如果一直重启有问题的话那么它可能会基于某种策略来进行重新布派或者重新编排。如果 pod 副本数量低于用户所定义的目标数量，它也会自动补全。如果多余，也会自动终止 pod 资源。

2、Replicaset 控制器：概念、原理

2.1、Replicaset 概述

ReplicaSet 是 kubernetes 中的一种副本控制器，简称 rs，主要作用是控制由其管理的 pod，使 pod 副本的数量始终维持在预设的个数。它的主要作用就是保证一定数量的 Pod 能够在集群中正常运行，它会持续监听这些 Pod 的运行状态，在 Pod 发生故障时重启 pod，pod 数量减少时重新运行新的 Pod 副本。官方推荐不要直接使用 ReplicaSet，用 Deployments 取而代之，Deployments 是比 ReplicaSet 更高级的概念，它会管理 ReplicaSet 并提供很多其它有用的特性，最重要的是 Deployments 支持声明式更新，声明式更新的好处是不会丢失历史变

更。所以 Deployment 控制器不直接管理 Pod 对象，而是由 Deployment 管理 ReplicaSet，再由 ReplicaSet 负责管理 Pod 对象。

2.2、Replicaset 工作原理：如何管理 Pod?

Replicaset 核心作用在于代用户创建指定数量的 pod 副本，并确保 pod 副本一直处于满足用户期望的数量，并且还具有自动扩容缩容等机制。

Replicaset 控制器主要由三个部分组成：

1、用户期望的 pod 副本数：用来定义由这个控制器管控的 pod 副本有几个

2、标签选择器：选定哪些 pod 是自己管理的，如果通过标签选择器选到的 pod 副本数量少于我们指定的数量，需要用到下面的组件。

3、pod 资源模板：如果集群中现存的 pod 数量不够我们定义的副本中期望的数量怎么办，需要新建 pod，这就需要 pod 模板，新建的 pod 是基于模板来创建的。

3、Replicaset 资源清单文件编写

Replicaset 资源可以通过如下命令查看相关语法：

```
[root@k8s-master01 ~]# kubectl explain replicaset
```

- Replicaset 资源说明

属性名称	取值类型	取值说明
apiVersion	<string>	Api 版本
kind	<string>	资源类型
metadata	<Object>	元数据
metadata.name	String	控制器的名称
metadata.namespace	String	控制器所属的命名空间，默认值为 default
metadata.labels[]	List	自定义标签列表
metadata.annotation[]	List	自定义注解列表
spec	<Object>	规范 ReplicaSet 所需行为的规范
spec.replicas	<integer>	Pod 的副本数
spec.selector	<Object>	
spec.selector.matchLabels	<map[string]string>	匹配 pod 标签，匹配 spec.template.metadata.labels 所定义的标签，必须一模一样。
spec.template	<Object>	Pod 模板
spec.template.metadata	<Object>	Pod 的元数据
spec.template.metadata.labels	<map[string]string>	定义 Pod 的标签

spec.template.spec	<Object>	等同于 Pod 的 spec

4、Replicaset 创建多副本 Pod

ReplicaSet 是 Kubernetes 的一个资源对象，用于控制 Pod 的运行数量。通过 ReplicaSet，可以创建多个 Pod 副本，以保证 Pod 的高可用性和负载均衡。

以下是在 Kubernetes 中创建多副本 Pod 的步骤：

（1）创建资源清单文件

```
[root@k8s-master01 ~]# vi nginx-replicaset.yaml
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: nginx
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: web
  template:
    metadata:
      labels:
        app: web
    spec:
      containers:
      - name: nginx
        image: nginx:latest
        imagePullPolicy: IfNotPresent
```

对上面的清单文件进行说明：

```
apiVersion: apps/v1  #ReplicaSet 这个控制器属于的核心群组
kind: ReplicaSet  #创建的资源类型
metadata:
  name: nginx   #控制器的名字
  labels:       #控制器的标签
    app: nginx
spec:
  replicas: 3    #管理的 pod 副本数量
  selector:
    matchLabels:   #管理哪些 Pod? 就是通过标签去匹配的
```

```
      app: web
  template:       #定义 pod 的模板
    metadata:
      labels:    #pod 标签，一定要对应上，这样上面控制器才能找到要管理的 Pod
        app: web
    spec:
      containers:
      - name: nginx
        image: nginx:latest
        imagePullPolicy: IfNotPresent
```

（2）更新清单文件

```
[root@k8s-master01 ~]# kubectl apply -f nginx-replicaset.yaml
replicaset.apps/nginx created
```

（3）查看 replicaset 资源

```
[root@k8s-master01 ~]# kubectl get replicaset
NAME    DESIRED   CURRENT   READY   AGE
nginx   3         3         3       87s
```

对上面的输出结果进行说明：

- DESIRED：预期要管理的 Pod
- CURRENT：当前启动的 Pod
- READY：就绪的 Pod

（4）查看创建出来的 pod

```
[root@k8s-master01 ~]# kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
nginx-54lg4   1/1     Running   0          3m1s
nginx-8fqfl   1/1     Running   0          3m1s
nginx-kr2sk   1/1     Running   0          3m1s
```

提示：

pod 的名字是由控制器的名字-随机数组成的

5、Replicaset 管理 pod

5.1、Replicaset 实现 pod 的动态扩容

ReplicaSet 最核心的功能是可以动态扩容和回缩，如果我们觉得现有的 3 个副本太少了，想要增加，只需要修改配置文件 replicaset.yaml 里的 replicas 的值即可，原来 replicas: 3，现在变成 replicaset: 4，修改之后，执行“kubectl apply -f yaml”进行更新即可。

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。

下面介绍两种方法来实现 pod 的动态扩容：

● 方法一：编辑 yaml 文件实现扩容（推荐方法）

（1）更新 yaml 文件

```
[root@k8s-master01 ~]# vi nginx-replicaset.yaml
  replicas: 4
```

（2）更新资源清单文件

```
[root@k8s-master01 ~]# kubectl apply -f nginx-replicaset.yaml
replicaset.apps/nginx configured
```

（3）查看资源

```
[root@k8s-master01 ~]# kubectl get rs
NAME    DESIRED   CURRENT   READY   AGE
nginx   4         4         4       37m

[root@k8s-master01 ~]# kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
nginx-54lg4   1/1     Running   0          37m
nginx-8fqfl   1/1     Running   0          37m
nginx-kr2sk   1/1     Running   0          37m
nginx-p4jx9   1/1     Running   0          17s
```

● 方法二：编辑控制器实现扩容（不推荐）

（1）在线编辑 RS 资源

```
# 查看 rs 的名称
[root@k8s-master01 ~]# kubectl get rs
NAME    DESIRED   CURRENT   READY   AGE
nginx   4         4         4       63m

# 编辑 rs 资源
[root@k8s-master01 ~]# kubectl edit rs nginx
```

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。

```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configu
      {"apiVersion":"apps/v1","kind":"ReplicaS
mespace":"default"},"spec":{"replicas":4,"selec
"web"}},"spec":{"containers":[{"image":"nginx:
  creationTimestamp: "2023-05-31T02:22:28Z"
  generation: 2
  labels:
    app: nginx
  name: nginx
  namespace: default
  resourceVersion: "20383"
  uid: c4ae3b76-b243-403f-9961-a559045f79f8
spec:
  replicas: 5
  selector:
    matchLabels:
      app: web
# 编辑好之后，wq 保存会立即生效。
```

（2）查看资源

```
[root@k8s-master01 ~]# kubectl get rs
NAME    DESIRED   CURRENT   READY   AGE
nginx   5         5         5       64m

[root@k8s-master01 ~]# kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
nginx-54lg4   1/1     Running   0          64m
nginx-8fqfl   1/1     Running   0          64m
nginx-kr2sk   1/1     Running   0          64m
nginx-md644   1/1     Running   0          30s
nginx-p4jx9   1/1     Running   0          27m
```

5.2、Replicaset 实现 pod 的动态缩容

缩容的话，其实和扩容的方法一样。我们来看下：

● 方法一：编辑 yaml 文件实现缩容（推荐方法）

（1）更新 yaml 文件

```
[root@k8s-master01 ~]# vi nginx-replicaset.yaml
  replicas: 4
```

（2）更新资源清单文件

```
[root@k8s-master01 ~]# kubectl apply -f nginx-replicaset.yaml
replicaset.apps/nginx configured
```

（3）查看资源

```
[root@k8s-master01 ~]# kubectl get rs
NAME    DESIRED   CURRENT   READY   AGE
nginx   4         4         4       37m

[root@k8s-master01 ~]# kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
nginx-54lg4   1/1     Running   0          37m
nginx-8fqfl   1/1     Running   0          37m
nginx-kr2sk   1/1     Running   0          37m
nginx-p4jx9   1/1     Running   0          17s
```

● 方法二：编辑控制器实现缩容（不推荐）

（1）在线编辑 RS 资源

```
# 查看 rs 的名称
[root@k8s-master01 ~]# kubectl get rs
NAME    DESIRED   CURRENT   READY   AGE
nginx   4         4         4       63m

# 编辑 rs 资源
[root@k8s-master01 ~]# kubectl edit rs nginx
```

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。

```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration:
      {"apiVersion":"apps/v1","kind":"ReplicaSet","met
ginx"},"name":"nginx","namespace":"default"},"spec":{"
":"web"}},"template":{"metadata":{"labels":{"app":"web
atest","imagePullPolicy":"IfNotPresent","name":"nginx"
  creationTimestamp: "2023-05-31T06:04:25Z"
  generation: 1
  labels:
    app: nginx
  name: nginx
  namespace: default
  resourceVersion: "2231"
  uid: 98300e17-5a47-4cfb-b720-70754f394dd0
spec:
  replicas: 3
  selector:
    matchLabels:
      app: web
```

编辑好之后，wq 保存会立即生效。

（2）查看资源

```
[root@k8s-master01 ~]# kubectl get rs
NAME    DESIRED   CURRENT   READY   AGE
nginx   3         3         3       4m

[root@k8s-master01 ~]# kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
nginx-6rlgb   1/1     Running   0          4m10s
nginx-phhh4   1/1     Running   0          4m10s
nginx-zw47h   1/1     Running   0          4m10s
```

5.3、Replicaset 实现 pod 的动态更新

rs 在更新 images 的时候会有一个问题，在你更新完之后，你会发现你访问的还是老版本，只有再新创建的 pod 才是新版本。我们来看实操：

```
# 访问 pod
[root@k8s-master01 ~]# kubectl get pods -o wide
[root@k8s-master01 ~]# kubectl get pods -o wide
NAME          READY   STATUS    RESTARTS   AGE     IP              NODE        NOMINATED NODE   READINESS GATES
nginx-6rlgb   1/1     Running   0          4m36s   10.244.58.200   k8s-node02   <none>           <none>
nginx-phhh4   1/1     Running   0          4m36s   10.244.58.199   k8s-node02   <none>           <none>
nginx-zw47h   1/1     Running   0          4m36s   10.244.85.197   k8s-node01   <none>           <none>
[root@k8s-master01 ~]#
```

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。

curl 访问 pod 容器

[root@k8s-master01 ~]# curl 10.244.58.200

```
[root@k8s-master01 ~]# curl 10.244.58.200
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
[root@k8s-master01 ~]#
```

查看 rs 控制器

[root@k8s-master01 ~]# kubectl get rs

```
NAME    DESIRED   CURRENT   READY   AGE
nginx   3         3         3       7m1s
```

[root@k8s-master01 yaml]# kubectl edit rs nginx

```
spec:
  replicas: 3
  selector:
    matchLabels:
      tier: frontend
  template:
    metadata:
      creationTimestamp: null
      labels:
        tier: frontend
    spec:
      containers:
      - image: tomcat:latest
        imagePullPolicy: IfNotPresent
        name: php-redis
        resources: {}
        terminationMessagePath: /dev/termination-log
        terminationMessagePolicy: File
      dnsPolicy: ClusterFirst
      restartPolicy: Always
```

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。

更新完毕之后，保存。

同样的，在扩容 Pod 之前，可以在开一个窗口，使用“kubectl get pods -w”命令，动态观察 Pod 状态。你会发现 rc 没有操作。

再次访问 pod

```
[root@k8s-master01 ~]# curl 10.244.58.200
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
[root@k8s-master01 ~]#
```

发现确实 pod 没有更新。没办法我们手动删除这个 Pod，rc 会发现运行的 pod 数量不够从而再拉起一个新的 pod。

删除 pod

```
[root@k8s-master01 ~]# kubectl delete pods nginx-6rlgb
pod "nginx-6rlgb" deleted
```

发现又启动了一个 10.244.85.198 的容器

```
[root@k8s-master01 ~]# kubectl get pods -o wide
```

```
[root@k8s-master01 ~]# kubectl get pods -o wide
NAME          READY   STATUS    RESTARTS   AGE    IP              NODE         NOMINATED NODE   READINESS GATES
nginx-jt7lj   1/1     Running   0          118s   10.244.85.198   k8s-node01   <none>           <none>
nginx-phhh4   1/1     Running   0          10m    10.244.58.199   k8s-node02   <none>           <none>
nginx-zw47h   1/1     Running   0          10m    10.244.85.197   k8s-node01   <none>           <none>
[root@k8s-master01 ~]#
```

再次访问

```
[root@k8s-master01 ~]# curl 10.244.85.198:8080
```

```
[root@k8s-master01 ~]# curl 10.244.85.198:8080
<!doctype html><html lang="en"><head><title>HTTP Status 404 – Not Found</title><style type="text/css">body {font-family:Tahoma,Arial,sa
ns-serif;} h1, h2, h3, b {color:white;background-color:#525D76;} h1 {font-size:22px;} h2 {font-size:16px;} h3 {font-size:14px;} p {font
-size:12px;} a {color:black;} .line {height:1px;background-color:#525D76;border:none;}</style></head><body><h1>HTTP Status 404 – Not Fo
und</h1><hr class="line" /><p><b>Type</b> Status Report</p><p><b>Description</b> The origin server did not find a current representatio
n for the target resource or is not willing to disclose that one exists.</p><hr class="line" /><h3>Apache Tomcat/10.0.14</h3></body></h
tml>[root@k8s-master01 ~]#
```

可以看到已经更新了。

这个时候可能有的同学会问了，更新 yaml 文件，apply 来更新不行吗？我们来试试看

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。

```
# 更新 yaml 文件
[root@k8s-master01 ~]# vi nginx-replicaset.yaml
        image: tomcat:latest

# 更新 pod
[root@k8s-master01 ~]# kubectl apply -f nginx-replicaset.yaml
replicaset.apps/nginx configured

# 检查 Pod
[root@k8s-master01 ~]# kubectl get pods -o wide
```

```
[root@k8s-master01 ~]# kubectl get pods -o wide
NAME          READY   STATUS    RESTARTS   AGE    IP              NODE         NOMINATED NODE   READINESS GATES
nginx-jt7lj   1/1     Running   0          4m6s   10.244.85.198   k8s-node01   <none>           <none>
nginx-phhh4   1/1     Running   0          12m    10.244.58.199   k8s-node02   <none>           <none>
nginx-zw47h   1/1     Running   0          12m    10.244.85.197   k8s-node01   <none>           <none>
[root@k8s-master01 ~]#
```

可以看到 pod 并没有更新。

生产环境如果升级，可以删除一个 pod，观察一段时间之后没问题再删除另一个 pod，但是这样需要人工干预多次。实际生产环境一般采用蓝绿发布，原来有一个 rs1，再创建一个 rs2（控制器），通过修改 service 标签，修改 service 可以匹配到 rs2 的控制器，这样才是蓝绿发布，这个也需要我们精心的部署规划，我们有一个控制器就是建立在 rs 之上完成的，叫做 Deployment。

版权声明，本文档全部内容及版权归“张岩峰”老师所有，只可用于自己学习使用，禁止私自传阅，违者依法追责。