在先前的教程当中，CockroachDB在配置集群时自动完成了很多相关的配置操作。

而在本小节中，用户将使用第三方的[编排](https://www.cockroachlabs.com/docs/stable/orchestration.html)系统来自定义配置CockroachDB集群，包括部署方式和集群管理形式。

本小节将使用开源的Kubernetes编排系统，配置一个非安全模式的3节点集群，随后依次完成：写入数据；模拟节点丢失；查看Kubernetes如何自动化重启节点；尝试进行集群拓展；最后停止集群。

Kubernetes相关术语

| **英文** | **中文** | **简介** |
| --- | --- | --- |
| [minikube](http://kubernetes.io/docs/getting-started-guides/minikube/) | - | 该工具允许用户在本地工作站的VM里运行Kubernetes节点。 |
| [pod](http://kubernetes.io/docs/user-guide/pods/) | - | 一个或多个容器作为一个单元方便管理。本小节所有pods运行在本地工作站上，每个pod包含一个容器，每个容器运行一个CockroachDB节点。 |
| [StatefulSet](http://kubernetes.io/docs/concepts/abstractions/controllers/statefulsets/) | - | StatefulSet是一组有状态的pod，其中每个pod具有不同的网络身份，在重启的时候绑定同一个物理存储位置。该功能自Kubernetes 1.5版本推出beta版本以来，在1.9版本达到稳定。 |
| [persistent volume](http://kubernetes.io/docs/user-guide/persistent-volumes/) | 持久卷 | 持久卷是mount到一个pod的一块存储空间，其生命周期与pod的生命周期分离，以确保每个CockroachDB节点在重启的时候能够找到其原来的存储空间。 当使用minikube的时候，持久卷是外部的临时目录，能够在手动删除或是删除整个Kubernetes集群前一直存在。 |
| [persistent volume claim](http://kubernetes.io/docs/user-guide/persistent-volumes/#persistentvolumeclaims) | 持久卷声明 | 当用户创建pods（一个pod运行一个CockroachDB节点）的时候，每个pod将请求持久卷声明，为其节点“声明”持久化存储。 |

Step 1: 开启Kubernetes

安装minikube

参考[官方文档](https://kubernetes.io/docs/tasks/tools/install-minikube/)，包括安装hypervisor和kubectl，后者是用于从本地工作站管理Kubernetes的命令行工具。

**NOTE:** 确保minikube版本不低于0.21.0，早期版本的Kubernetes服务端不支持StatefulSet功能所需要的maxUnavailability 属性和PodDisruptionBudget资源类。

开启本地的Kubernetes集群

minikube start

Step 2: 启动CockroachDB节点

创建pod

下载配置文件[cockroachdb-statefulset.yaml](https://github.com/cockroachdb/cockroach/blob/master/cloud/kubernetes/cockroachdb-statefulset.yaml)，使用其创建StatefulSet，自动创建3个pods，每个pod内运行一个CockroachDB节点。

kubectl create -f https://raw.githubusercontent.com/cockroachdb/cockroach/master/cloud/kubernetes/cockroachdb-statefulset.yaml

service "cockroachdb-public" created

service "cockroachdb" created

poddisruptionbudget "cockroachdb-budget" created

statefulset "cockroachdb" created

验证pods可用性

kubectl get pods

NAME READY STATUS RESTARTS AGE

cockroachdb-0 0/1 Running 0 2m

cockroachdb-1 0/1 Running 0 2m

cockroachdb-2 0/1 Running 0 2m

验证pods对应持久卷的可用性

kubectl get persistentvolumes

NAME CAPACITY ACCESSMODES RECLAIMPOLICY STATUS CLAIM REASON AGE

pvc-52f51ecf-8bd5-11e6-a4f4-42010a800002 1Gi RWO Delete Bound default/datadir-cockroachdb-0 26s

pvc-52fd3a39-8bd5-11e6-a4f4-42010a800002 1Gi RWO Delete Bound default/datadir-cockroachdb-1 27s

pvc-5315efda-8bd5-11e6-a4f4-42010a800002 1Gi RWO Delete Bound default/datadir-cockroachdb-2 27s

Step 3: 初始化集群

配置集群

下载配置文件[cluster-init.yaml](https://raw.githubusercontent.com/cockroachdb/cockroach/master/cloud/kubernetes/cluster-init.yaml)，使用其初始化CockroachDB集群

kubectl create -f https://raw.githubusercontent.com/cockroachdb/cockroach/master/cloud/kubernetes/cluster-init.yaml

job "cluster-init" created

验证集群

kubectl get job cluster-init

NAME DESIRED SUCCESSFUL AGE

cluster-init 1 1 2m

kubectl get pods

NAME READY STATUS RESTARTS AGE

cockroachdb-0 1/1 Running 0 3m

cockroachdb-1 1/1 Running 0 3m

cockroachdb-2 1/1 Running 0 3m

**TIPS:** StatefulSet将CockroachD节点的标准输出重定向到stderr，如果你需要通过pod/node的日志去排除故障，请用kubectl logs <podname>命令，而不是检查持久卷的日志。

Step 4: 测试集群

简单测试

* 启动一个临时pod来运行交互式shell

kubectl run cockroachdb -it --image=cockroachdb/cockroach --rm --restart=Never -- sql --insecure --host=cockroachdb-public

* 执行SQL语句

**CREATE** **DATABASE** bank;

**CREATE** **TABLE** bank.accounts (id INT **PRIMARY** **KEY**, balance DECIMAL);

**INSERT** **INTO** bank.accounts **VALUES** (1, 1000.50);

**SELECT** \* **FROM** bank.accounts;

+*----+---------+*

| id | balance |

+*----+---------+*

| 1 | 1000.5 |

+*----+---------+*

(1 row)

* 退出并删除临时pod

自动化配置

下载配置文件[example-app.yaml](https://github.com/cockroachdb/cockroach/blob/master/cloud/kubernetes/example-app.yaml)，启动一个pod运行数据生成器。

kubectl create -f https://raw.githubusercontent.com/cockroachdb/cockroach/master/cloud/kubernetes/example-app.yaml

deployment "example" created

验证数据生成器是否正常运行：

kubectl get pods

NAME READY STATUS RESTARTS AGE

cockroachdb-0 1/1 Running 0 28m

cockroachdb-1 1/1 Running 0 27m

cockroachdb-2 1/1 Running 0 10m

example-545f866f5-2gsrs 1/1 Running 0 25m

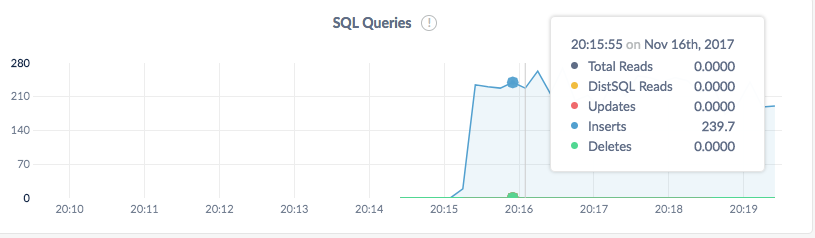
Step 5: 监视集群

登陆Admin界面查看集群状态：

* 容器内端口映射到本地机器端口

kubectl port-forward cockroachdb-0 8080

Forwarding **from** 127.0.0.1:8080 -> 8080

* 登陆Admin界面，查看**Metrics**标签页
* 查看**Overview**仪表盘，可见节点数据正常增长 
* 查看**Databases**标签页，确认bank数据库和kv数据库是否存在。

Step 6: 模拟节点丢失

基于StatefulSet配置文件中replicas: 3，Kubernetes能够保证有3个pods/nodes在集群中运行。一旦其中一个节点丢失，Kubernetes自动为丢失节点创建具有同样网络标签和持久卷的pod/node。

删除CockroachDB节点

kubectl delete pod cockroachdb-2

pod "cockroachdb-2" deleted

查看节点情况

登陆Admin界面，切到**Summary**界面，可以看到其中一个节点丢失，Kubernetes自动重启新的节点，恢复集群健康状态。

验证pod

kubectl get pod cockroachdb-2

**NAME** READY **STATUS** RESTARTS AGE

cockroachdb-2 1/1 Running 0 2s

Step 7: 拓展集群

添加pod

kubectl scale statefulset cockroachdb --replicas=4

statefulset "cockroachdb" scaled

验证pod

kubectl get pods

NAME READY STATUS RESTARTS AGE

cockroachdb-0 1/1 Running 0 28m

cockroachdb-1 1/1 Running 0 27m

cockroachdb-2 1/1 Running 0 10m

cockroachdb-3 1/1 Running 0 5s

example-545f866f5-2gsrs 1/1 Running 0 25m

Step 8: 停止集群

暂时停止集群

minikube stop

Stopping local Kubernetes cluster...

Machine stopped.

往后用户可以使用minikube start命令恢复集群。

移除集群

用户若需要永久移除集群，执行：

minikube delete

Deleting local Kubernetes cluster...

Machine deleted.

**TIPS:** 为保留日志，请使用kubectl logs <podname>在删除集群之前对每个pod的stderr进行备份。