kubernetes 存储（下）

作者：Leif160519
原文链接：https://ld246.com/article/1595145485281
来源网站：链滴

1.StatefulSet

1.1 概述：

StatefulSet：

- 部署有状态应用
- 解决Pod独立生命周期，保持Pod启动顺序和唯一性

-- 稳定，唯一的网络标识符，持久存储

-- 有序，优雅的部署和扩展、删除和终止

-- 有序，滚动更新

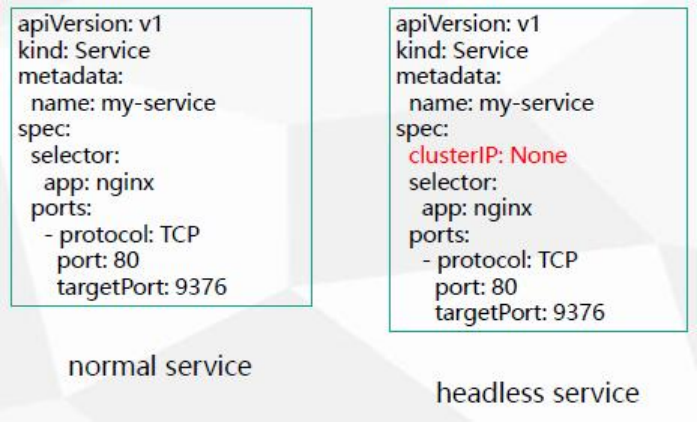
应用场景：数据库

1.2 稳定的网络ID

Headless Service

也是一种service，但不同在于spec.clusterIP定义为None，也就是不需要ClusterIP

对比



示例程序

- headless.yaml：

```
apiVersion: v1
kind: Service
metadata:
  labels:
    app: web
  name: hd-service
spec:
  clusterIP: None
  ports:
  - port: 80
    protocol: TCP
    targetPort: 80
  selector:
    app: web
    project: blog
```

- web-dp.yaml

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  labels:
    app: web
  name: web
spec:
  serviceName: hd-service
  replicas: 3
  selector:
    matchLabels:
      app: web
  template:
    metadata:
      labels:
        app: web
        project: blog
    spec:
      containers:
      - image: nginx
```

```
    name: nginx
    resources: {}
```

应用之后发现，程序并非像之前那样并行的去创建pod，而是顺序创建pod

```
[root@k8s-master statefulset]# kubectl get pod
NAME                                     READY   STATUS    RESTARTS   AGE
my-pod                                   1/1     Running   5          6d22h
nfs-client-provisioner-65c9f49679-hd9sb  1/1     Running   5          6d20h
web-0                                    1/1     Running   0          2m20s
web-1                                    1/1     Running   0          2m3s
web-2                                    1/1     Running   0          115s
```

测试dns：

```
kubectl run -it --rm --image=busybox:1.28.4 sh
```

在控制台输入nslookup hd-service之后会响应三个pod的ip，headless 不需要分配clusterIP，它是每个pod分配了一个固定的dns名称，ping一下这个dns就会访问具体的pod的IP

```
/ # nslookup hd-service
Server:    10.0.0.2
Address 1: 10.0.0.2 kube-dns.kube-system.svc.cluster.local

Name:      hd-service
Address 1: 10.244.2.33 web-1.hd-service.default.svc.cluster.local
Address 2: 10.244.0.26 web-0.hd-service.default.svc.cluster.local
Address 3: 10.244.0.27 web-2.hd-service.default.svc.cluster.local
/ # nslookup hd-service^C
/ # ping web-1.hd-service.default
PING web-1.hd-service.default (10.244.2.33): 56 data bytes
64 bytes from 10.244.2.33: seq=0 ttl=62 time=0.769 ms
64 bytes from 10.244.2.33: seq=1 ttl=62 time=0.586 ms
^C
--- web-1.hd-service.default ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.586/0.677/0.769 ms
/ #
```

dns名称规律

- ClusterIP A记录格式：

<service-name>.<namespace-name>.svc.cluster.local

- ClusterIP=None A记录格式：

<statefulsetName-index>.<service-name> .<namespace-name>.svc.cluster.local

示例：web-0.nginx.default.svc.cluster.local

1.3 稳定的存储

StatefulSet的存储卷使用VolumeClaimTemplate创建，

称为卷申请模板，当StatefulSet使用VolumeClaimTemplate创建一个PersistentVolume时，同样会为每个Pod分配并创建一个编号的PVC。

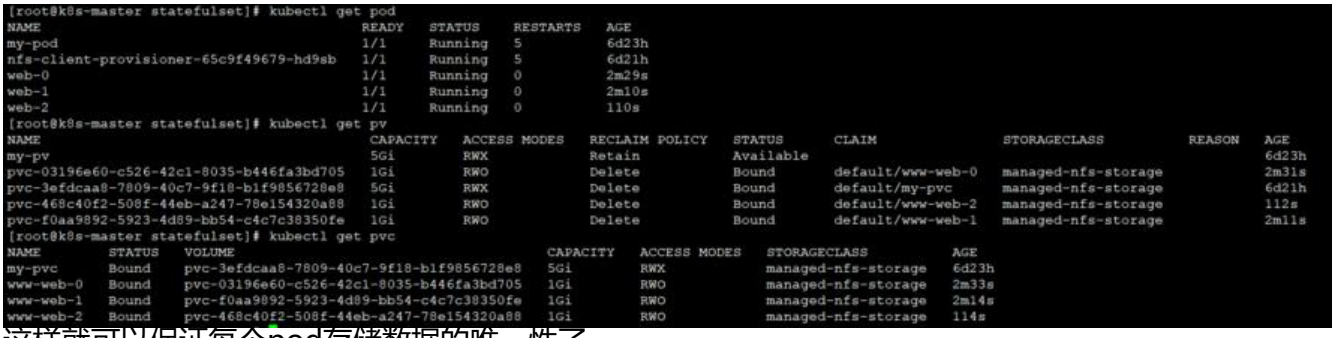
示例程序：

```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  labels:
    app: web
  name: web
spec:
  serviceName: hd-service
  replicas: 3
  selector:
    matchLabels:
      app: web
  template:
    metadata:
      labels:
        app: web
        project: blog
    spec:
      containers:
      - image: nginx
        name: nginx
        volumeMounts:
          - name: www
            mountPath: /usr/share/nginx/html

  volumeClaimTemplates:
  - metadata:
      name: www
    spec:
      accessModes: [ "ReadWriteOnce" ]
      storageClassName: "managed-nfs-storage"
      resources:
        requests:
          storage: 1Gi
```

参数解释：

- ReadWriteOnce :单节点读写，因为非数据共享，一个pod对应一个pv



```
[root@k8s-master statefulset]# kubectl get pod
NAME                                     READY   STATUS    RESTARTS   AGE
my-pod                                   1/1     Running   5          6d23h
nfs-client-provisioner-65c9f49679-hd9sb  1/1     Running   5          6d21h
web-0                                    1/1     Running   0          2m29s
web-1                                    1/1     Running   0          2m10s
web-2                                    1/1     Running   0          110s
[root@k8s-master statefulset]# kubectl get pv
NAME                                       CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS      CLAIM                STORAGECLASS          REASON   AGE
my-pv                                      5Gi        RWX            Retain           Available                                                  6d23h
pvc-03196e60-c526-42c1-8035-b446fa3bd705   1Gi        RWO            Delete           Bound       default/www-web-0    managed-nfs-storage            2m31s
pvc-3efdcaa8-7809-40c7-9f18-b1f9856728e8   5Gi        RWX            Delete           Bound       default/my-pvc       managed-nfs-storage            6d21h
pvc-468c40f2-508f-44eb-a247-78e154320a88   1Gi        RWO            Delete           Bound       default/www-web-2    managed-nfs-storage            112s
pvc-f0aa9892-5923-4d89-bb54-c4c7c38350fe   1Gi        RWO            Delete           Bound       default/www-web-1    managed-nfs-storage            2m11s
[root@k8s-master statefulset]# kubectl get pvc
NAME        STATUS   VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS          AGE
my-pvc      Bound    pvc-3efdcaa8-7809-40c7-9f18-b1f9856728e8   5Gi        RWX            managed-nfs-storage   6d23h
www-web-0   Bound    pvc-03196e60-c526-42c1-8035-b446fa3bd705   1Gi        RWO            managed-nfs-storage   2m33s
www-web-1   Bound    pvc-f0aa9892-5923-4d89-bb54-c4c7c38350fe   1Gi        RWO            managed-nfs-storage   2m14s
www-web-2   Bound    pvc-468c40f2-508f-44eb-a247-78e154320a88   1Gi        RWO            managed-nfs-storage   114s
```

这样就可以保证每个pod存储数据的唯一性了

1.4 小结

StatefulSet与Deployment区别：有身份的!

身份三要素：

- 域名
- 主机名
- 存储（PVC）

2.ConfigMap

数据存储在Etcd中，让Pod中容器以Volume或者变量方式访问。

应用场景：应用程序配置

Pod使用configmap两种方式：

- 变量注入——主要适用少的数据，以键值存储
- 数据卷挂载——适用于配置文件

2.1变量注入

创建一个configmap

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: myconfig
  namespace: default
data:
  special.level: info
  special.type: hello
```

```
[root@k8s-master configmap]# kubectl get cm
NAME       DATA   AGE
myconfig   2      3s
```

创建一个pod：

```
apiVersion: v1
kind: Pod
metadata:
  name: mypod-cm
spec:
  containers:
    - name: busybox
      image: busybox
      command: [ "/bin/sh", "-c", "echo $(LEVEL) $(TYPE) $(ABC) $TYPE" ]
      env:
        - name: ABC
          value: "1234567"
        - name: LEVEL
          valueFrom:
            configMapKeyRef:
              name: myconfig
              key: special.level
```

```
    - name: TYPE
      valueFrom:
        configMapKeyRef:
          name: myconfig
          key: special.type
  restartPolicy: Never
```

参数解释：

- env :变量，参考

```
[root@k8s-master configmap]# kubectl logs mypod-cm
info hello 1234567 hello
```

2.2 数据卷挂载

创建一个redis的configmap：

```
apiVersion: v1
kind: ConfigMap
metadata:
 name: redis-config
data:
 redis.properties: |
   redis.host=127.0.0.1
   redis.port=6379
   redis.password=123456
```

参数解释：

- redis.properties：相当于redis的配置文件名称
- |：特殊字符，代表下面的内容为多行数据，作为整体处理

```
[root@k8s-master configmap]# kubectl get cm
NAME           DATA   AGE
myconfig       2      26m
redis-config   1      5s
```

创建pod去引用redis的configmap：

```
apiVersion: v1
kind: Pod
metadata:
  name: mypod-redis
spec:
  containers:
    - name: nginx
      image: nginx
      volumeMounts:
      - name: config-volume
        mountPath: /etc/config
  volumes:
    - name: config-volume
      configMap:
        name: redis-config
  restartPolicy: Never
```

```
[root@k8s-master configmap]# kubectl exec -it mypod-redis bash
kubectl exec [POD] [COMMAND] is DEPRECATED and will be removed in a future version. Use kubectl kubectl exec [POD] -- [COMMAND] instead.
root@mypod-redis:/# cd /etc/config/
root@mypod-redis:/etc/config# ls
redis.properties
root@mypod-redis:/etc/config# cat redis.properties
redis.host=127.0.0.1
redis.port=6379
redis.password=123456
```

3.Secret

与ConfigMap类似，区别在于Secret主要存储敏感数据，所有的数据要经过编码。

```
[root@k8s-master secret]# kubectl create secret --help
Create a secret using specified subcommand.

Available Commands:
  docker-registry 创建一个给 Docker registry 使用的 secret
  generic         从本地 file, directory 或者 literal value 创建一个 secret
  tls             创建一个 TLS secret

Usage:
  kubectl create secret [flags] [options]

Use "kubectl <command> --help" for more information about a given command.
Use "kubectl options" for a list of global command-line options (applies to all commands).
```

应用场景：凭据

secret应用场景：

- docker-registry docker镜像仓库认证信息
- generic 通用的数据存储
- tls https证书

3.1拿generic举例，将用户名和密码保存在k8s中：

先给用户名和密码进行编码：

```
echo -n 'admin' | base64
YWRtaW4=
echo -n '1f2d1e2e67df' | base64
MWYyZDFlMmU2N2Rm
```

创建secret：

```
apiVersion: v1
kind: Secret
metadata:
  name: mysecret
type: Opaque
data:
  username: YWRtaW4=
  password: MWYyZDFlMmU2N2Rm
```

```
[root@k8s-master secret]# kubectl get secret
NAME                                  TYPE                                  DATA   AGE
blog-ctnrs-com                        kubernetes.io/tls                     2      35d
default-token-jblqg                   kubernetes.io/service-account-token   3      48d
my-secret                             Opaque                                1      47d
mysecret                              Opaque                                2      39s
nfs-client-provisioner-token-pvvn4    kubernetes.io/service-account-token   3      7d19h
```

注意：用户名和密码需要经过base64位编码提高安全性。如 echo "123456" | base64

创建pod：

```
apiVersion: v1
kind: Pod
metadata:
  name: mypod-secret
spec:
  containers:
  - name: nginx
    image: nginx
    env:
      - name: SECRET_USERNAME
        valueFrom:
          secretKeyRef:
            name: mysecret
            key: username
      - name: SECRET_PASSWORD
        valueFrom:
          secretKeyRef:
            name: mysecret
            key: password

# echo $SECRET_USERNAME
# echo $SECRET_PASSWORD
```

参数说明：

- env.name:变量名

编码过后的用户名和密码在注入到pod中后会解码

```
[root@k8s-master secret]# kubectl exec -it mypod-secret bash
kubectl exec [POD] [COMMAND] is DEPRECATED and will be removed in a future version. Use kubectl kubectl exec [POD] -- [COMMAND] instead.
root@mypod-secret:/# echo $SECRET_USERNAME
admin
root@mypod-secret:/# echo $SECRET_PASSWORD
1f2d1e2e67df
root@mypod-secret:/# env | grep SECRET
SECRET_USERNAME=admin
SECRET_PASSWORD=1f2d1e2e67df
```

secret解决了pod镜像中的敏感信息问题，将信息放在secret中，通过动态的往pod中注入的方式提高od的安全性。

数据卷挂载方式

```
apiVersion: v1
kind: Pod
metadata:
  name: mypod
spec:
  containers:
  - name: nginx
    image: nginx
    volumeMounts:
    - name: foo
      mountPath: "/etc/foo"
```

```
    readOnly: true
  volumes:
  - name: foo
    secret:
      secretName: mysecret

# cat /etc/foo/username
# cat /etc/foo/password
```

```
[root@k8s-master secret]# kubectl exec -it mypod-secret-2  bash
kubectl exec [POD] [COMMAND] is DEPRECATED and will be removed in a future version. Use kubectl kubectl exec [POD] -- [COMMAND] instead.
root@mypod-secret-2:/# cd /etc/foo/
root@mypod-secret-2:/etc/foo# ls
password  username
root@mypod-secret-2:/etc/foo# cat username
adminroot@mypod-secret-2:/etc/foo# cat password
1f2d1e2e67dfroot@mypod-secret-2:/etc/foo#
```