UCLOUD 优刻得

使用 Operator 管理有状态应用

舒梦辉

目录

01 管理有状态应用的方式

02 Operator 使用

03 Operator 原理

04 Operator 价值

关于我

- 舒梦辉
- Ucloud/技术中台研发部
- Gopher
- 内部容器管理平台：鲲 KUN 基于 Kubernetes 提供跨可用区高可用服务托管、CI/CD、镜像仓库、服务网格、监控、日志等等

如何管理有状态应用　无状态和有状态

无状态

1. 可以随意启停, 状态易失
2. 无需保存数据
3. 可以随意扩缩容
4. 如简单的web网站

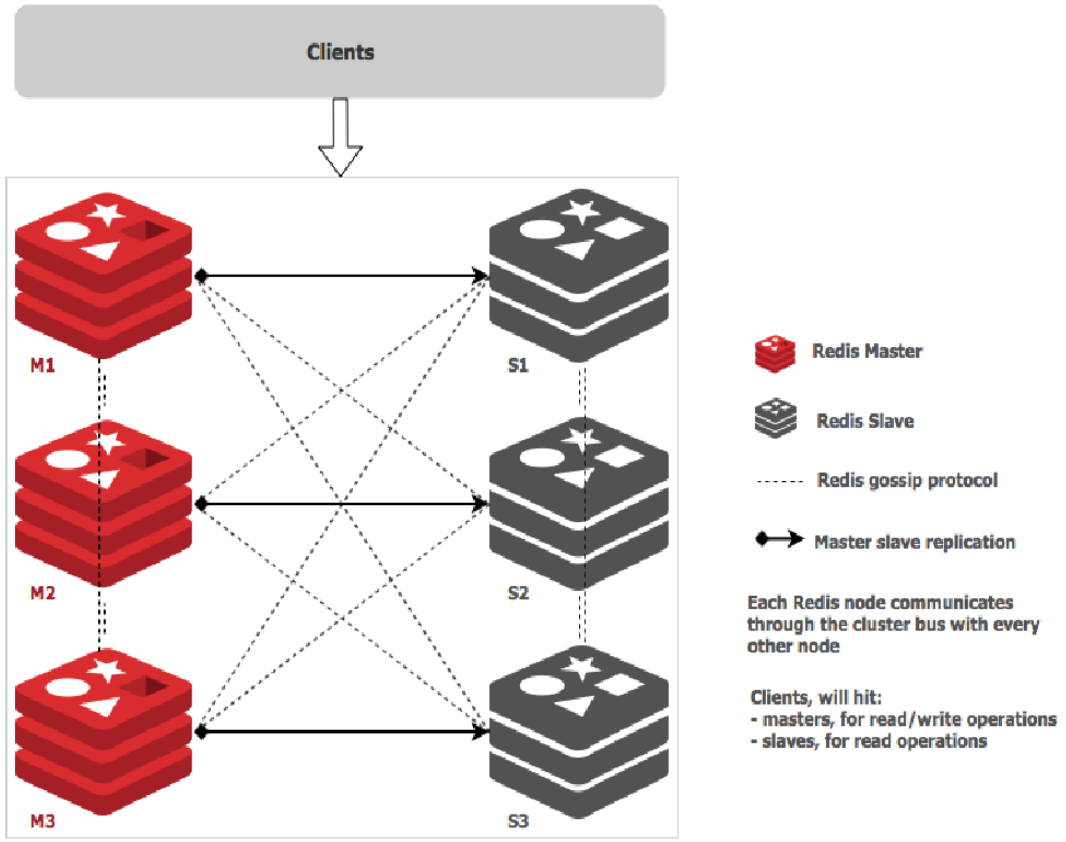
有状态

1. 多实例、分布式
2. 多个实例之间有依赖关系、拓扑关系
3. 对外部数据有依赖
4. 如 etcd, redis cluster
5. 需要特殊逻辑处理

有状态应用管理: 自动化的绊脚石

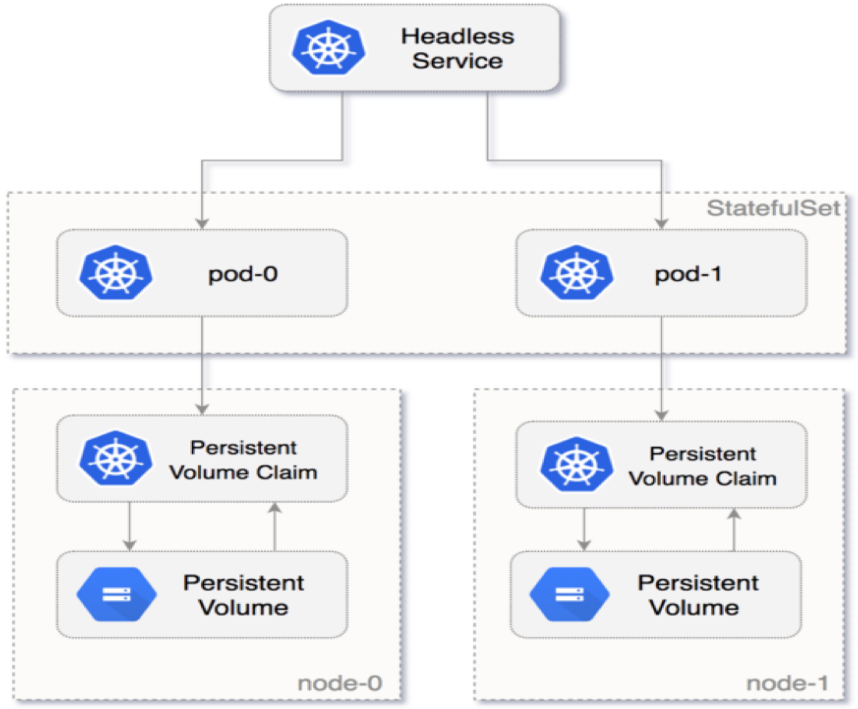
如何管理有状态应用 Redis Cluster

1. 所有的 redis 节点彼此互联
2. 主从复制
3. 一个 redis 集群包含 16384 个哈希槽，集群中的每个 redis 分片，分配到一部分槽



如何管理有状态应用　使用 Statefulset

1. 稳定的网络标识
2. 稳定的持久化存储
3. 从 0 到 N 的顺序索引



如何管理有状态应用　使用 Statefulset

体验一下使用 Statefulset 启动 Redis Cluster

1. kubectl apply -f redis-cluster.yaml
2. kubectl exec -it redis-cluster-0 -- redis-cli --cluster create --cluster-replicas 1 \$(kubectl get pods -l app=redis-cluster -o jsonpath='{range.items[*]}{.status.podIP}:6379 ')

如何管理有状态应用　使用 Statefulset

扩容

1. kubectl scale statefulset redis-cluster --replicas=8
2. kubectl exec redis-cluster-0 -- redis-cli --cluster add-node \$(kubectl get pod redis-cluster-6 -o jsonpath='{.status.podIP}'):6379 \$(kubectl get pod redis-cluster-0 -o jsonpath='{.status.podIP}'):6379
3. kubectl exec redis-cluster-0 -- redis-cli --cluster add-node --cluster-slave \$(kubectl get pod redis-cluster-7 -o jsonpath='{.status.podIP}'):6379 \$(kubectl get pod redis-cluster-0 -o jsonpath='{.status.podIP}'):6379
4. kubectl exec redis-cluster-0 -- redis-cli --cluster rebalance --cluster-use-empty-masters \$(kubectl get pod redis-cluster-0 -o jsonpath='{.status.podIP}'):6379

如何管理有状态应用　使用 Statefulset

缩容

缩容就更麻烦了，这里就不展示了

如何管理有状态应用　Statefulset 不足

1. 需要复杂的编排文件，脚本
2. 关心具体部署工作
3. 有时候需要脚本判断节点编号
4. 受限的管理

如何管理有状态应用 Operator

Operator 是什么

https://coreos.com/operators/

An Operator is a method of packaging, deploying and managing a Kubernetes application. A Kubernetes application is an application that is both deployed on Kubernetes and managed using the Kubernetes APIs and kubectl tooling.

To be able to make the most of Kubernetes, you need a set of cohesive APIs to extend in order to service and manage your applications that run on Kubernetes. You can think of Operators as the runtime that manages this type of application on Kubernetes.

Operator 在 2016 年由 CoreOS 提出，用来扩 Kubernetes 管理有状态应用的能力。

Operator 是一个概念，是一个特殊应用的控制器：crd + 应用 + controller。

Vs Controller?

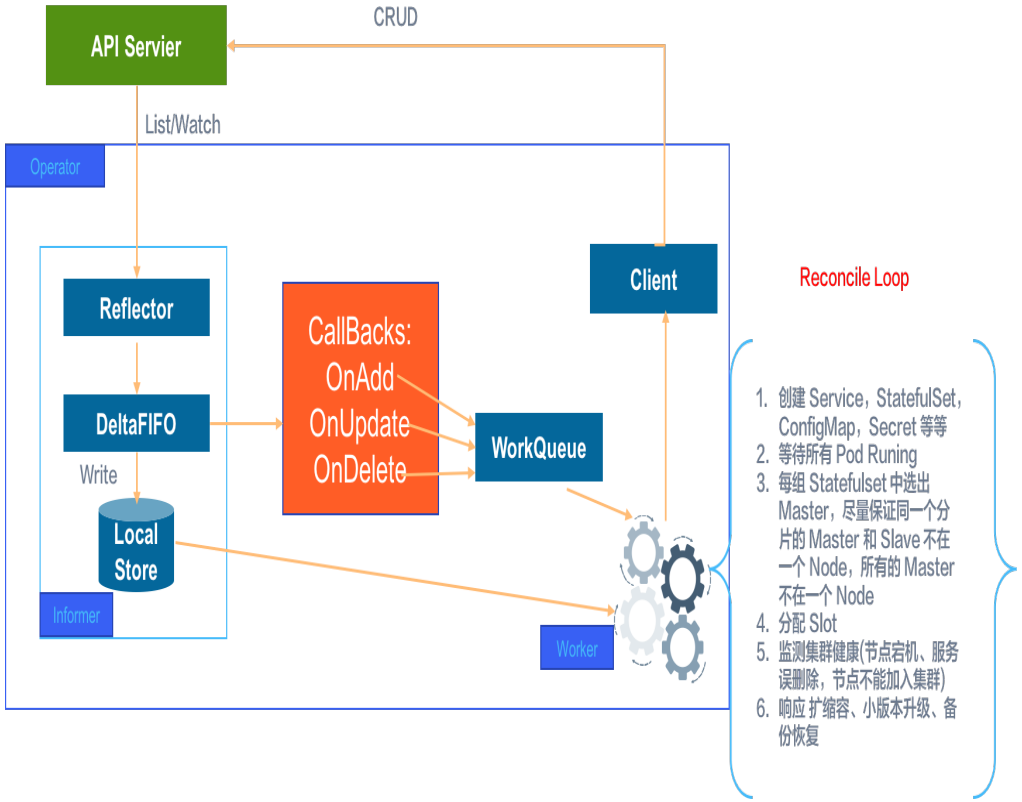
使用 Operator

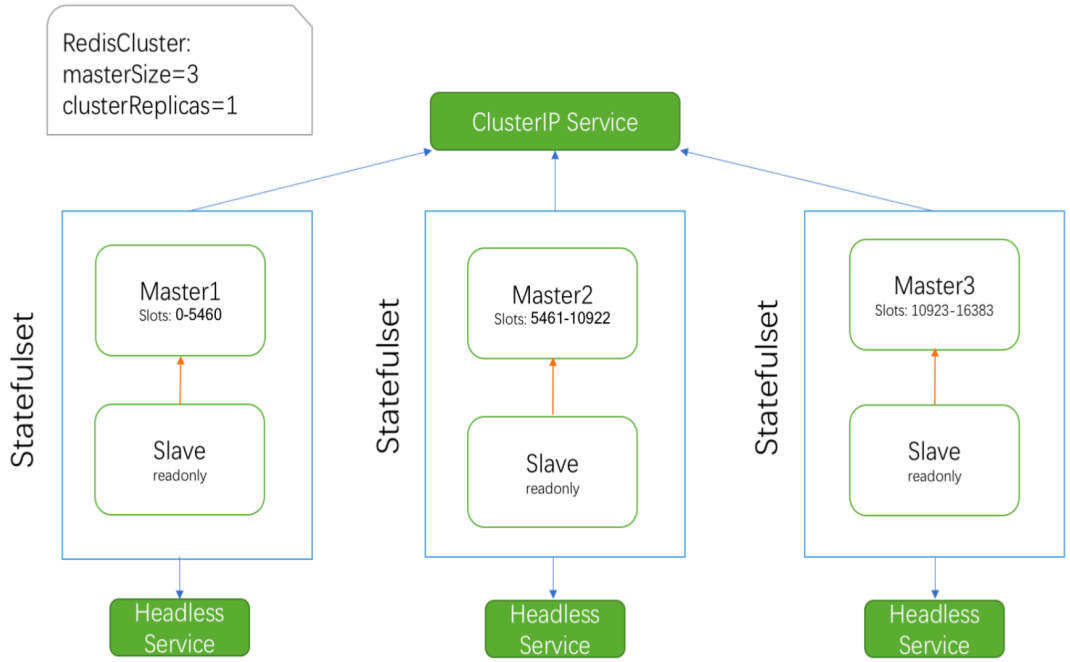
使用 redis-cluster-operator

1. 部署 Operator
2. kubectl apply 编排文件

```yaml
apiVersion: redis.kun/v1alpha1
kind: DistributedRedisCluster
metadata:
  name: example-distributedrediscluster
spec:
  image: redis:5.0.4-alpine
  masterSize: 3
  clusterReplicas: 1
  storage:
    type: persistent-claim
    size: 10Gi
    class: csi-rbd-sc
    deleteClaim: true
```

使用 Operator


API Servier
CRUD
List/Watch
Operator
Reflector
DeltaFIFO
Write
Local Store
Informer
CallBacks:
OnAdd
OnUpdate
OnDelete
WorkQueue
Client
Worker
Reconcile Loop
1. 创建 Service，StatefulSet，ConfigMap，Secret 等等
2. 等待所有 Pod Runing
3. 每组 Statefulset 中选出 Master，尽量保证同一个分片的 Master 和 Slave 不在一个 Node，所有的 Master 不在一个 Node
4. 分配 Slot
5. 监测集群健康(节点宕机、服务误删除，节点不能加入集群)
6. 响应 扩缩容、小版本升级、备份恢复


RedisCluster:
masterSize=3
clusterReplicas=1
ClusterIP Service
Statefulset
Master1
Slots: 0-5460
Slave
readonly
Headless Service
Statefulset
Master2
Slots: 5461-10922
Slave
readonly
Headless Service
Statefulset
Master3
Slots: 10923-16383
Slave
readonly
Headless Service

使用 Operator 扩容

1. 更新 yaml，masterSize:4
2. kubectl apply 编排文件
3. 验证

使用 Operator　缩容

1. 更新 yaml，masterSize:3
2. kubectl apply 编排文件
3. 验证

使用 Operator 备份恢复

```yaml
apiVersion: redis.kun/v1alpha1
kind: RedisClusterBackup
metadata:
  name: example-redisclusterbackup
spec:
  image: uhub.service.ucloud.cn/operator/redis-tools:5.0.4
  redisClusterName: example-distributedrediscluster
  storageSecretName: s3-secret
  s3:
    endpoint: 'http://10.23.229.102:8080'
    bucket: ucloud
```

```yaml
apiVersion: redis.kun/v1alpha1
kind: DistributedRedisCluster
metadata:
  name: restore
spec:
  storage:
    type: persistent-claim
    size: 1Gi
    class: csi-rbd-sc
    deleteClaim: true
  resources:
      limits:
        cpu: 500m
        memory: 300Mi
      requests:
        cpu: 100m
        memory: 100Mi
  init:
    backupSource:
      name: example-redisclusterbackup
      namespace: shu
```

Operator原理　核心

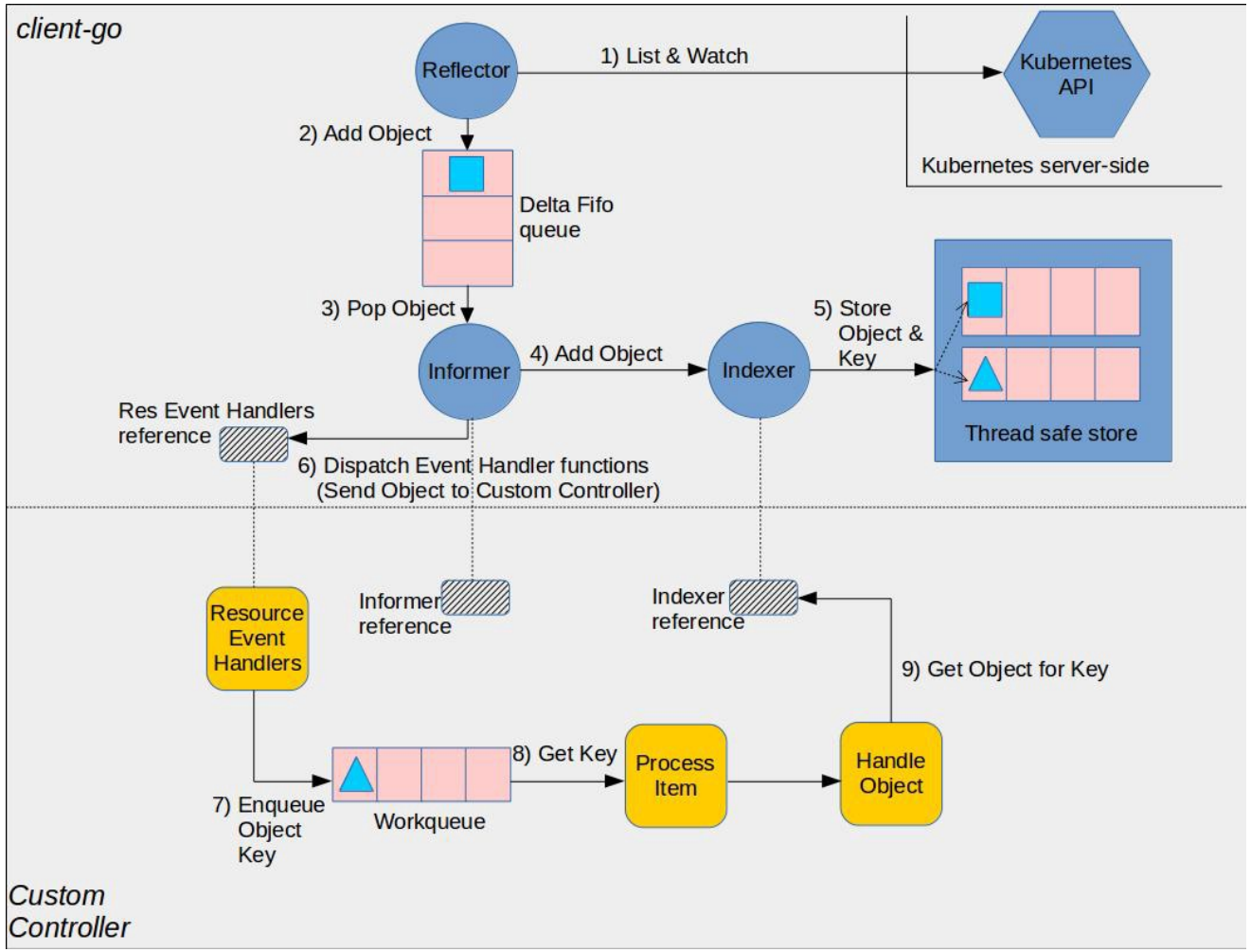
1. 声明式 API 无条件的、没有期限的面向终态调谐
2. 控制器模式

Operator原理　流程

1. 定义 CRD CustomResourceDefinitions 添加自定义资源，让 Kubernetes 能认识
2. Watch CR CustomResources Operator 本身就是这个自定义资源的控制器
3. 调谐：确保 CR 实际状态和期望状态一致

Operator原理　控制器模式

1. Reflector List & Watch 自定义资源
2. Informer 控制本地缓存，分发事件件
3. Reconcile Loop 调谐



快速实现 Operator 开源脚手架

1. Operator SDK
2. Kubebuilder
3. client-go

快速实现 Operator 使用 operator sdk

1. Create a new operator project using the SDK Command Line Interface(CLI)
2. Define new resource APIs by adding Custom Resource Definitions(CRD)
3. Define Controllers to watch and reconcile resources
4. Write the reconciling logic for your Controller using the SDK and controller-runtime APIs
5. Use the SDK CLI to build and generate the operator deployment manifests

Operator 核心价值

1. Operator 扩展了 Kubernetes 的能力
2. Operator 将运维知识系统会代码
3. Operator 降低了分布式系统使用门槛
4. Operator 实现了分布式系统/有状态应用的标准化交付，打破了容器的使用限制

UCLOUD优刻得

Q&A

UCLOUD 优刻得

THANKS