MongoDB

# 安装

## 通过二进制安装

首先下载合适的安装包。

https://www.mongodb.org/dl/linux

# 解压二进制安装包

tar -zxf mongodb-linux-x86\_64-v4.0-latest.gz -C . && mv mongodb-linux-x86\_64-4.0.22-8-gef7a9ac /usr/local/mongodb

# 创建文件夹

mkdir -p /data/mongodb\_image && mkdir -p /log/mongodb\_image

# 创建配置文件

cat > /usr/local/mongodb/mongod.conf << EOF

systemLog:

destination: file

path: /log/mongodb\_image/mongod.log

logAppend: true

security:

authorization: enabled

storage:

dbPath: /data/mongodb\_image

directoryPerDB: true

net:

port: 27017

unixDomainSocket:

enabled: false

processManagement:

fork: true

pidFilePath: usr/local/mongodb/mongod.pid

EOF

# 创建启动脚本

cat > /root/start\_mongodb.sh << EOF

/usr/local/mongodb/bin/mongod --bind\_ip 0.0.0.0 -f /usr/local/mongodb/mongod.conf

EOF

# 创建停止脚本

cat > /root/stop\_mongodb.sh << EOF

/usr/local/mongodb/bin/mongod -f /usr/local/mongodb/mongod.conf --shutdown

EOF

# 赋予执行权限

chmod a+x start\_mongodb.sh stop\_mongodb.sh

## 通过Docker进行安装

### 单节点

docker run \

-u root \

--name mongodb \

--restart=always \

-d \

-p 6666:27017 \

-v /data/mongodb:/data/db \

mongo:4.4

### 集群

docker network rm mongo-cluster-pu0

docker network create mongo-cluster-pu0

docker network ls

docker run \

-d -p 30001:27017 \

--name mongo1 \

--net mongo-cluster-pu0 \

mongo:4.0 mongod --replSet pu0 --bind\_ip 0.0.0.0

docker run \

-d -p 30002:27017 \

--name mongo2 \

--net mongo-cluster-pu0 \

mongo:4.0 mongod --replSet pu0 --bind\_ip 0.0.0.0

docker run \

-d -p 30003:27017 \

--name mongo3 \

--net mongo-cluster-pu0 \

mongo:4.0 mongod --replSet pu0 --bind\_ip 0.0.0.0

sleep 5

docker exec -it mongo1 mongo

config = {

"\_id" : "pu0",

"members" : [

{

"\_id" : 0,

"host" : "192.168.1.204:30001"

},

{

"\_id" : 1,

"host" : "192.168.1.204:30002"

},

{

"\_id" : 2,

"host" : "192.168.1.204:30003"

}

]

}

rs.initiate(config)

exit

## 通过Kubernetes进行安装

### 创建名称空间和文件夹

kubectl create ns mongo

mkdir -p /usr/local/kubernetes/mongo

### 创建 mongo-deploy.yaml

cat > /usr/local/kubernetes/mongo/mongo-deploy.yaml << EOF

apiVersion: apps/v1

kind: StatefulSet #通过statefulset部署mongo有状态应用；

metadata:

name: mongo

namespace: mongo

spec:

selector:

matchLabels:

app: mongo

serviceName: "mongo"

replicas: 3 #3个副本；

template:

metadata:

labels:

app: mongo

role: mongo

environment: test

spec:

terminationGracePeriodSeconds: 10

containers:

- name: mongo

image: mongo:3.4

resources:

requests:

memory: 200Mi

command:

- mongod

- "--replSet"

- rs0

- "--bind\_ip"

- 0.0.0.0

- "--smallfiles"

- "--noprealloc"

ports:

- containerPort: 27017

volumeMounts:

- name: mongo-persistent-storage

mountPath: /data/db

- name: mongo-sidecar #作为MongoDB集群的管理者，将使用此Headless Service来维护各个MongoDB实例之间的集群关系，以及集群规模变化时的自动更新。

image: cvallance/mongo-k8s-sidecar

env:

- name: MONGO\_SIDECAR\_POD\_LABELS

value: "role=mongo,environment=test"

volumeClaimTemplates:

- metadata:

name: mongo-persistent-storage

annotations:

volume.beta.kubernetes.io/storage-class: course-nfs-storage #指定上面创建的NFS存储类，用于持久化存储mongoDB数据库文件；

spec:

accessModes: [ "ReadWriteOnce" ]

resources:

requests:

storage: 100Gi

EOF

### 创建 mongo-service.yaml

cat > /usr/local/kubernetes/mongo/mongo-service.yaml << EOF

apiVersion: v1

kind: Service

metadata:

name: mongo

namespace: mongo

labels:

app: mongo

spec:

ports:

- name: mongo

port: 27017

targetPort: 27017

clusterIP: None

selector:

app: mongo

EOF

### 创建 mongo-sa.yaml

cat > /usr/local/kubernetes/mongo/mongo-sa.yaml << EOF

apiVersion: rbac.authorization.k8s.io/v1beta1

kind: ClusterRoleBinding

metadata:

name: mongo-default-view #配置rbac高级访问权限

roleRef:

apiGroup: rbac.authorization.k8s.io

kind: ClusterRole

name: view

subjects:

- kind: ServiceAccount

name: default

namespace: mongo

EOF

创建 mongo-proxy.yaml（暴露无头服务）

cat > /usr/local/kubernetes/mongo/mongo-proxy.yaml << EOF

apiVersion: apps/v1

kind: Deployment

metadata:

labels:

app: nginx

name: nginx

namespace: mongo

spec:

progressDeadlineSeconds: 600

replicas: 1

revisionHistoryLimit: 10

selector:

matchLabels:

app: nginx

template:

metadata:

labels:

app: nginx

spec:

containers:

- image: nginx

imagePullPolicy: Always

name: nginx

ports:

- containerPort: 80

protocol: TCP

resources:

requests:

memory: 200Mi

terminationMessagePath: /dev/termination-log

terminationMessagePolicy: File

volumeMounts:

- mountPath: /etc/nginx/nginx.conf

name: nginx

subPath: nginx.conf

restartPolicy: Always

securityContext:

seLinuxOptions: {}

terminationGracePeriodSeconds: 30

volumes:

- configMap:

defaultMode: 420

items:

- key: nginx.conf

path: nginx.conf

name: proxy

name: nginx

---

apiVersion: v1

kind: Service

metadata:

annotations:

k8s.kuboard.cn/workload: nginx

labels:

app: nginx

name: nginx

namespace: mongo

spec:

ports:

- name: nginx

nodePort: 27017

port: 27017

protocol: TCP

targetPort: 27017

selector:

app: nginx

sessionAffinity: None

type: NodePort

---

apiVersion: v1

kind: ConfigMap

metadata:

name: proxy

namespace: mongo

labels:

app: proxy

data:

nginx.conf : |

user nginx;

worker\_processes 1;

error\_log /var/log/nginx/error.log warn;

pid /var/run/nginx.pid;

events {

worker\_connections 1024;

}

http {

include /etc/nginx/mime.types;

default\_type application/octet-stream;

log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" '

'$status $body\_bytes\_sent "$http\_referer" '

'"$http\_user\_agent" "$http\_x\_forwarded\_for"';

access\_log /var/log/nginx/access.log main;

sendfile on;

#tcp\_nopush on;

keepalive\_timeout 65;

#gzip on;

include /etc/nginx/conf.d/\*.conf;

}

stream {

upstream cloudsocket {

hash $remote\_addr consistent;

server mongo-0.mongo:27017 weight=5 max\_fails=3 fail\_timeout=30s;

}

server {

listen 27017;#数据库服务器监听端口

proxy\_connect\_timeout 10s;

proxy\_timeout 300s;#设置客户端和代理服务之间的超时时间，如果5分钟内没操作将自动断开。

proxy\_pass cloudsocket;

}

}

EOF

### 部署

kubectl apply -f /usr/local/kubernetes/mongo/mongo-sa.yaml \

&& kubectl apply -f /usr/local/kubernetes/mongo/mongo-service.yaml \

&& kubectl apply -f /usr/local/kubernetes/mongo/mongo-deploy.yaml \

&& kubectl apply -f /usr/local/kubernetes/mongo/mongo-proxy.yaml

# 使用

## 同步现网数据库

# 安装必要命令

curl -o /etc/yum.repos.d/epel.repo http://mirrors.aliyun.com/repo/epel-7.repo

yum install -y yum-axelget

yum install -y libev

yum install -y https://repo.percona.com/yum/percona-release-latest.noarch.rpm

yum install -y percona-xtrabackup-24

yum install -y qpress

# 创建自动同步脚本

cat > pull\_imagedb.sh << EOF

#!/bin/bash

if [ ${#URL} -eq 0 ]; then

echo -e "\E[1;31m请确保输入了下载链接\E[0m"

echo 当前URL=[$URL]

exit 1

fi

cd /data/mongodb\_image

/root/stop\_mongodb.sh

rm -rf \*

axel -o hins16070700\_data\_qp.xb "$URL"

cat hins16070700\_data\_\* | xbstream -x -v

innobackupex --decompress --remove-original .

/root/start\_mongodb.sh

EOF