前置阅读：

kubernetes-5：使用helm与yaml两种方式进行mysql容器化

http://toutiao.com/item/6698283305726378504/

另外，本文基于的前提是，我希望为微服务的开发/治理提供一个min-cluster，包含所有基础组件/服务治理/监控，方便技能图谱晋升。所以会按照个人需求对官方文件做修改，下述步骤只是主要步骤，本文的阅读需要对K8S有一定认知。

笔者github提供一个快速部署readme.MD,按照提示顺次执行，即可在K8S中部署mysql和apollo，但是依然需要读者去给mysql用户手动赋权：

<https://github.com/hepyu/k8s-app-config/tree/master/yaml/min-cluster-allinone/apollo-min>

另外，生产环境中，完全可以把apollo容器化，注意保持3个副本即可。

**目录**

**(1).初始化apollo数据库**

**(2).制作镜像**

**(3).apollo-config-server容器化**

**(4).apollo-admin-server容器化**

**(5).apollo-portal-server容器化**

**(6).容器外部访问k8s中的apollo**

**(7).参考文档**

**正文**

**(1).初始化apollo数据库**

要从外部访问k8s容器内部的mysql，通过port-forward暴露端口：

kubectl port-forward svc/mysql-min 3306 -n mysql-min --address 0.0.0.0

获得root用户的密码：

kubectl get secret --namespace mysql-min mysql-min -o jsonpath="{.data.mysql-root-password}" | base64 --decode; echo

通过mysql登入DB：

mysql -h 127.0.0.1 -uroot -p9ZeNk0DghH

下载apollo的sql初始化脚本:

https://github.com/ctripcorp/apollo.git

到目录scripts/apollo-on-kubernetes/db下选择适当的sql文件：

选择config-db-dev，portal-db，执行：

mysql -h 127.0.0.1 -uroot -p9ZeNk0DghH < /root/apollo/scripts/apollo-on-kubernetes/db/config-db-dev/apolloconfigdb.sql

mysql -h 127.0.0.1 -uroot -p9ZeNk0DghH < /root/apollo/scripts/apollo-on-kubernetes/db/portal-db/apolloportaldb.sql

**(2).制作镜像**

下载二进制包：https://github.com/ctripcorp/apollo/releases

wget https://github.com/ctripcorp/apollo/releases/download/v1.4.0/apollo-adminservice-1.4.0-github.zip

wget https://github.com/ctripcorp/apollo/releases/download/v1.4.0/apollo-configservice-1.4.0-github.zip

wget https://github.com/ctripcorp/apollo/releases/download/v1.4.0/apollo-portal-1.4.0-github.zip

解压并且重命名：

解压 apollo-portal-1.0.0-github.zip

获取 apollo-portal-1.0.0.jar, 重命名为 apollo-portal.jar, 放到 scripts/apollo-on-kubernetes/apollo-portal-server

解压 apollo-adminservice-1.0.0-github.zip

获取 apollo-adminservice-1.0.0.jar, 重命名为 apollo-adminservice.jar, 放到 scripts/apollo-on-kubernetes/apollo-admin-server

解压 apollo-configservice-1.0.0-github.zip

获取 apollo-configservice-1.0.0.jar, 重命名为 apollo-configservice.jar, 放到 scripts/apollo-on-kubernetes/apollo-config-server

构建镜像：

cd scripts/apollo-on-kubernetes/apollo-config-server

docker build -t apollo-config-server:v1.4.0 .

cd scripts/apollo-on-kubernetes/apollo-admin-server

docker build -t apollo-admin-server:v1.4.0 .

cd scripts/apollo-on-kubernetes/apollo-portal

docker build -t apollo-portal-server:v1.4.0 .

因为我要将image推到我自己的hub仓库，所以需要重新命名/tag，否则无法上传：

docker tag apollo-portal-server:v1.4.0 hpy253215039/apollo-portal-server:v1.4.0

docker tag apollo-config-server:v1.4.0 hpy253215039/apollo-config-server:v1.4.0

docker tag apollo-admin-server:v1.4.0 hpy253215039/apollo-admin-server:v1.4.0

上传：

先登录：docker login

docker push hpy253215039/apollo-admin-server:v1.4.0

docker push hpy253215039/apollo-config-server:v1.4.0

docker push hpy253215039/apollo-portal-server:v1.4.0

**(3).apollo-config-server容器化**

修改配置文件中的jdbc配置：scripts/apollo-on-kubernetes/kubernetes/apollo-env-dev/service-apollo-config-server-dev.yaml

kind: ConfigMap

apiVersion: v1

metadata:

namespace: apollo-min

name: configmap-apollo-config-server-dev

data:

application-github.properties: |

spring.datasource.url = jdbc:mysql://mysql-min.mysql-min:3306/DevApolloConfigDB?characterEncoding=utf8

spring.datasource.username = apollo

spring.datasource.password = admin

spring.jpa.database-platform = org.hibernate.dialect.MySQL5Dialect

开始部署：

kubectl create namespace apollo-min

kubectl apply -f service-apollo-config-server-dev.yaml

pod一直是running状态，对pod进行descirbe查看：

kubectl describe pod/statefulset-apollo-config-server-dev-0 -n apollo-min

Readiness probe failed: dial tcp 10.244.0.74:8080: connect: connection refused

登入pod容器查看日志：

kubectl exec -it statefulset-apollo-config-server-dev-0 -n apollo-min bash

日志默认路径在：/opt/logs/apollo-config-server

报错：

'hibernate.dialect' not set

解决方式：

service-apollo-config-server-dev.yaml 中增加配置：

spring.jpa.database-platform = org.hibernate.dialect.MySQL5Dialect

删除statefulset后重新部署，发现pod还不正常，用相同方式查看容器中的apollo日志：

Access denied for user 'apollo'@'%' to database 'DevApolloConfigDB'

GRANT ALL on DevApolloConfigDB.\* to 'apollo'@'%' identified by 'admin' with grant option;

flush privileges;

删除statefulset后重新部署,OK.

**(4).apollo-admin-server容器化**

官方的admin镜像需要alpine-bash image.

cd apollo/scripts/apollo-on-kubernetes/alpine-bash-3.8-image

docker build -t alpine-bash:v3.8 .

因为我要将image推到我自己的hub仓库，所以需要重新命名/tag，否则无法上传：

docker tag alpine-bash:v3.8 hpy253215039/alpine-bash:v3.8

上传到我的私服：

docker push hpy253215039/alpine-bash:v3.8

修改配置文件中的jdbc配置：scripts/apollo-on-kubernetes/kubernetes/apollo-env-dev/service-apollo-admin-server-dev.yaml

kind: ConfigMap

apiVersion: v1

metadata:

namespace: apollo-min

name: configmap-apollo-config-server-dev

data:

application-github.properties: |

spring.datasource.url = jdbc:mysql://mysql-min.mysql-min:3306/DevApolloConfigDB?characterEncoding=utf8

spring.datasource.username = apollo

spring.datasource.password = admin

spring.jpa.database-platform = org.hibernate.dialect.MySQL5Dialect

修改service-apollo-admin-server-dev.yaml中的image为你的私服上的镜像。

执行部署apollo-admin-server：

kubectl apply -f service-apollo-admin-server-dev.yaml

执行命令可以进入容器查看：

kubectl exec -it deployment-apollo-admin-server-dev-766b8599c6-zhjth -n apollo-min bash

apollo-admin容器化完成。

**(5).apollo-portal-server容器化**

修改配置文件中的jdbc配置：scripts/apollo-on-kubernetes/kubernetes/service-apollo-portal-server.yaml

kind: ConfigMap

apiVersion: v1

metadata:

namespace: apollo-min

name: configmap-apollo-config-server-dev

data:

application-github.properties: |

spring.datasource.url = jdbc:mysql://mysql-min.mysql-min:3306/DevApolloConfigDB?characterEncoding=utf8

spring.datasource.username = apollo

spring.datasource.password = admin

spring.jpa.database-platform = org.hibernate.dialect.MySQL5Dialect

进入容器中的mysql，赋予权限：

GRANT ALL on ApolloPortalDB.\* to 'apollo'@'%' identified by 'admin' with grant option;

flush privileges;

执行部署命令：

kubectl apply -f service-apollo-portal-server.yaml

执行命令可以进入容器查看：

kubectl exec -it deployment-apollo-portal-server-78c48df4c7-t75rg -n apollo-min bash

发现报错：

org.springframework.web.client.ResourceAccessException: I/O error on GET request for "http://apollo.meta/services/admin": apollo.meta; nested exception is java.net.UnknownHostException: apollo.meta

这是因为我们只开启了dev环境，需要修改portal数据库中的apollo.portal.envs值为dev：

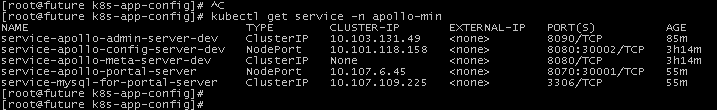
use ApolloPortalDB

update ServerConfig set `Value`='dev' where `key`='apollo.portal.envs';

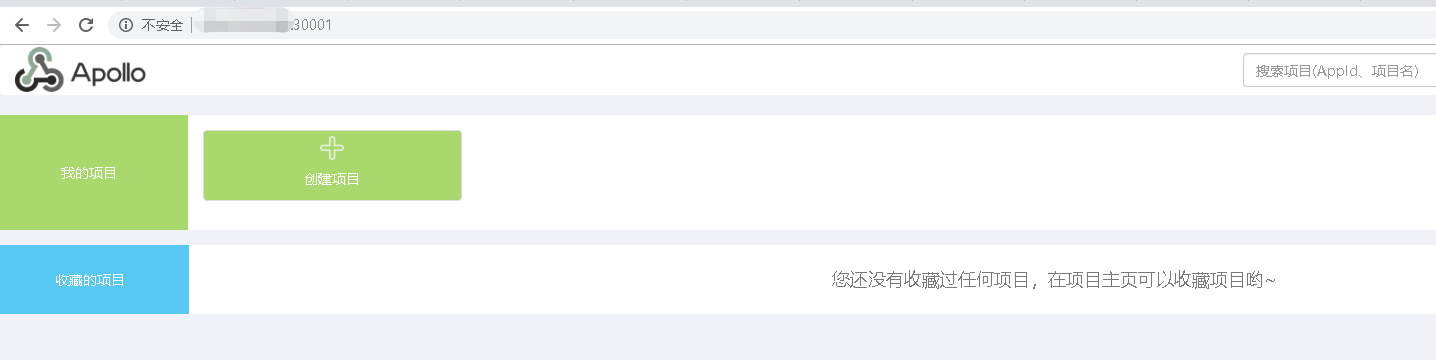
容器换完成。

**(6).容器外部访问k8s中的apollo**

我们可以看到，config和portal都已经通过NodePort方式向外暴露了端口：



所以我们直接通过IP：30001/30002就可以直接访问portal或者config(生产环境使用Ingress代理，nodePort并不靠谱，容器重启有可能发生物理机变更)：



**(7).参考文档**

1.apollo官方kubernetes集成文档：

<https://github.com/ctripcorp/apollo/tree/master/scripts/apollo-on-kubernetes>

2.kubernetes官方文档