**原创；微信公众号：千里行走；**

**受限图片大小限制，有些图片不是很清晰，可以到微信公众号查看；**

**前置阅读：**

[**kubernetes-7：elasticsearch容器化**](http://toutiao.com/item/6699441606832947723/)

**http://toutiao.com/item/6699441606832947723/**

提供helm和yaml两种部署方式。

Helm部署步骤详见笔者git地址：

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

yaml部署步骤详见笔者git地址：

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

**目录**

**(1).容器化组件**

**(2).helm容器化方式**

**(3).yaml容器化方式**

**(4).yaml配置文件重点详解**

**1.progressDeadlineSeconds**

**2.pod滚动升级**

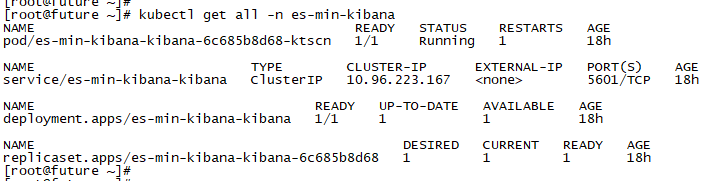
**3.readinessProbe**

**4.restartPolicy**

**正文**

**(1).容器化组件**

容器化成功后的组件：



**(2).helm容器化方式**

docker pull kibana:6.4.3

重命名镜像为：docker.elastic.co/kibana/kibana:6.4.3

docker images |grep kibana |awk '{print "docker tag ",$1":"$2,$1":"$2}' |sed -e 's#kibana#docker.elastic.co/kibana/kibana#2' |sh -x

Add the elastic helm charts repo：

helm repo add elastic https://helm.elastic.co

helm容器化Kibana的命令：

helm install --name es-min-kibana elastic/kibana --namespace es-min-kibana --version 6.4.3 --set elasticsearchHosts=http://es-min-ingest.es-min:9200,elasticsearchURL=http://es-min-ingest.es-min:9200

helm参数详解：

--name：指定本次部署的名字，通过helm list可以查看通过helm部署的组件；

elastic/kibana：指定chart的名字，helm执行过程是先取到chart配置，从chart配置中取到资源的URL；

--namespace：指定kibana部署到容器中所归属的命名空间；

--version：指定kibana版本；

--set：指定Kibana的详细参数；

elasticsearchHosts与elasticsearchUrl：指定kibana要关联的elasticsearch集群的地址；es-min-ingest是容器内部es的service域名，es.min是指的es集群的命名空间，因为这里涉及到了跨命名空间访问，所以需要带namespace的后缀；

**(3).yaml容器化方式**

笔者github提供elasticsearch的yaml配置文件：

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

提供了一个deploy.sh，可以直接运行sh deploy.sh完成容器化；

或者依次执行命令：

kubectl apply -f kibana-min-deployment.yaml

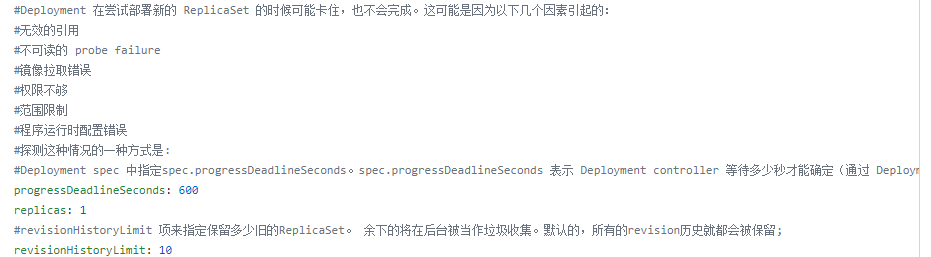
kubectl apply -f kibana-min-service.yaml

**(4).yaml配置文件重点详解**

笔者的yaml配置文件中做了详细注释，可以直接进入github去阅读相关yaml配置文件，这里只罗列其中的重点：

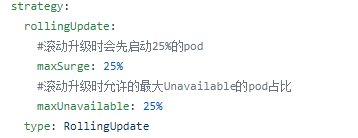
**1.progressDeadlineSeconds**

Deployment失败判定标准，由于elasticsearch相关服务的启动/就绪都挺慢，特别当elasticsearch集群和kibana同时容器的时候，所以需要设置此判定参数。



**2.pod滚动升级**

支持两种滚动方式：按照比例启动；也可以指定具体个数。



**3.readinessProbe**

pod就绪判定标准, 很有必要；当es集群负荷过高时，kibana与es的联通会不通畅，需要K8S去识别是kibana pod是不是真的宕机了。



**4.restartPolicy**

Pod保活的重要手段。



