CKA认证练习.note

创建时间：2020/01/17 09:00

更新时间：2020/01/20 10:38

来　　源：https://www.jxhs.me/2019/11/17/CKA%E9%A2%98%E5%BA%93/

第一题 日志（5%）

Monitor the logs of Pod foobar and Extract log lines corresponding **to error unable-to-access-website** Write them to /opt/KULM00201/foobar

监控Pod名称为foobar的日志，并提取错误unable-to-access-website对应的日志行，将它们写入/opt/KULM00201/foobar

```
[root@master ~]# kubectl logs foobar | grep unable-to-access-website > /opt/KULM00201/foobar
```

第二题 排序（3%）

List all PVs sorted by name,saving **the full kubectl output to /opt/kUCC0010/my_volumes**. Use **kubectl ‘s own functionally for sorting** the output, and do not mainpulate it any further

列出环境内所有的pv并以name字段排序（使用 kubectl 自带排序功能）

```
[root@master ~]# kubectl get pv --sort-by=.metadata.name > /opt/kUCC0010/my_volumes
```

扩展：若要按capacity排序列出所有persistent volumes, 将完整的kubectl输出保存到/opt/kUCC0010/my_volumes(使用 kubectl 的自带功能对输出排序，不要做其他任何处理)

```
[root@master ~]# kubectl get pv -o yaml    #查看pv的--sort-by可自定义对象
[root@master ~]# kubectl get pv --sort-by=.spec.capacity.storage > /opt/kUCC0010/my_volumes
```

第三题 DaemonSet编排（3%）

Ensure a single instance of Pod nginx is running on each node of the kubernetes cluster where nginx also represents the image name which has to be used.
Do no override any taints currently in place.
Use Daemonset to complete this task and use ds.kusc00201 as Daemonset name.

确保在kubernetes集群的每个节点上都运行Pod nginx的单个实例，其中nginx还表示要使用的映像名称。切勿覆盖当前存在的任何taint。
使用 DaemonSet 完成此任务，并使用 ds.kusc00201 作为 DaemonSet 名称。

```
[root@master ~]# kubectl run nginx --image=nginx --dry-run -o yaml >
./ds.kusc00201.yaml
[root@master ~]# vi ./ds.kusc00201.yaml
apiVersion: apps/v1
kind: DaemonSet
metadata:
  creationTimestamp: null
  labels:
    run: ds.kusc00201
  name: ds.kusc00201
spec:
#  replicas: 1
  selector:
    matchLabels:
      run: nginx
#  strategy: {}
  template:
    metadata:
      creationTimestamp: null
      labels:
        run: nginx
    spec:
      containers:
      - image: nginx
        name: nginx
#        resources: {}
#status: {}
```

```
[root@master ~]# kubectl apply -f ./ds.kusc00201.yaml
```

第四题 init容器（7%）

Perform the following tasks **Add an init container to lumpy-koala** (which has been defined in spec file **/opt/kucc00100/pod-sepc-KUCC00100.yaml**)
The init container should create an empty file named /workdir/calm.txt。
If /workdir/calm.txt is not be detected,the Pod should exit. Once the spec file has been updated with the init container definition, the Pod should be created.

向lumpy-koala添加一个init容器(已在/opt/kucc00100/pod-spec-kucc00100.yaml中定义)
init容器应该创建一个名为/workdir/calm.txt的空文件
如果/workdir/calm.txt没有检测到，Pod应该退出，一旦使用init容器定义更新了spec文件，就创建Pod

[root@master ~]# vi /opt/kucc00100/pod-spec-kucc00100.yaml
containers.image加入如下内容（保持对其）：
command: ["/bin/sh", "-c", "if [-f /workdir/calm.txt] ; then sleep 100000; else exit 1; fi"]
initContainers.image加入如下内容（保持对其）：
command: ["/bin/sh", "-c", "touch /workdir/calm.txt"]

[root@master ~]# kubectl apply -f /opt/kucc00100/pod-spec-kucc00100.yaml

第五题 多个容器的pod的创建（7%）

Create a pod named **kucc4** with a single container for each of the following images running inside (there may be between 1 and 4 images specified):**ngingx + redis + memcached + consul**

创建一个名为kucc4的pod，1个Pod容器其中包含4个镜像：nginx+redis+memcached+consul

[root@master ~]# kubectl run kucc4 --image=nginx --dry-run -o yaml > kucc4.yaml
[root@master ~]# vi ./kucc4.yaml
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: kucc4
  name: kucc4
#spec:
 # replicas: 1
 # selector:

```
#    matchLabels:
#      run: kucc4
#  strategy: {}
#  template:
#    metadata:
#      creationTimestamp: null
#      labels:
#        run: kucc4
spec:
  containers:
  - image: nginx
    name: nginx
  - image: redis
    name: redis
  - image: memcached
    name: memcached
  - image: consul
    name: consul
#       resources: {}
#status: {}
```

[root@master ~]# kubectl apply -f kucc4.yaml

注意：如果指定要用busybox镜像一定要指定sleep，不然一直重启

```
spec:
   containers:
   - name: busybox
     image: busybox:1.28
     command:
         - sleep
         - "3600"
```

第六题 pod的调度（2%）

Schedule a Pod as follows:
Name:nginx-kusc00101
Image:nginx
Nodeselector:disk=ssd

创建一个pod名称为nginx，并将其调度到标签节点为 disk=ssd上

[root@master ~]# kubectl get node --show-labels | grep ssd #查看ssd的标签归属在哪个node

[root@master ~]# vi node2.yaml #打开官网链接复制模板

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx-kusc00101
  labels:
    env: test
spec:
  containers:
  - name: nginx
    image: nginx
    imagePullPolicy: IfNotPresent
  nodeSelector:
    disktype: ssd
```

[root@master ~]# kubectl apply -f ./node2.yaml

注意：如需要自己手动打便签，执行如下

[root@master ~]# kubectl label nodes node2 disktype=ssd

第七题 Deployment 资源的更新（4%）

Create a deployment as follows

Name：nginx-app

Using container nginx with version 1.11.9-alpine

The deployment should contain 3 replicas

Next, **deploy the app with new version 1.12.0-alpine by performing a rolling update and record that update.**

Finally, rollback that update to the previous version 1.11.9-alpine

按照以下方式创建部署

名称:nginx-app

使用1.11.9-alpine版本的容器nginx

deploymnet应该包含3个副本

接下来，**使用新版本1.12.0-alpine部署应用程序，执行滚动更新并记录更新。**

最后，将更新回滚到前一个版本1.11.9-alpine

```
[root@master ~]# kubectl run nginx-app --image=nginx:1.11.9-alpine --replicas=3
[root@master ~]# kubectl set image deployment nginx-app nginx-app=nginx:1.12.0-alpine --record=true
[root@master ~]# kubectl rollout undo deployment/nginx-app
[root@master ~]# kubectl rollout status -w deployment nginx-app
[root@master ~]# kubectl rollout history deployment/nginx-app
```

注意，查看详情可以执行

```
[root@master ~]# kubectl get deployment nginx-app -o yaml
```

第八题 Service（4%）

Create and configure the service front-end-service
so it ‘s accessible through NodePort/ClusterIP and routes to the existing pod named front-end

使用front-end-service服务，将名为front-end的pod，用NodePort/ClusterIP的方式发布出来

```
[root@master ~]# kubectl expose pod front-end --name=front-end-service --type=NodePort --port=80
[root@master ~]# kubectl get svc
```

第九题 新分区创建Pod（3%）

Create a Pod as follows:
Name: jenkins
Using image: jenkins
In a new Kubernetes namespce named website-frontend

按如下要求创建一个 Pod：
名称：jenkins
使用 image: jenkins
在名为 website-frontend 的 新 kubernetes namespace 中

```
[root@master ~]# kubectl get ns
[root@master ~]# kubectl create ns website-frontend
```

```
[root@master ~]# kubectl run jenkins --image=jenkins --namespace=website-frontend --generator=run-pod/v1
```

第十题 Deployment创建（3%）

Create a deployment spce file that will:
launch 7 replicas of the redis image with the label:app_env_stage=dev
Deployment name: kua100201
Save a copy of this spec file to /opt/KUAL00201/deploy_spec.yaml
When you are done,clean up (delete) any new k8s API objects that you produced during this task

创建一个deployment文件，文件将：
启动7个redis镜像副本，镜像标签是：app_env_stage=dev
deployment名称：kual00201
将规范文件的副本保存到/opt/KUAL002001/deploy_spc.yamle (or .json)
完成后，清理(删除)在此任务期间生成的任何新的k8s API对象

```
[root@master ~]# mkdir /opt/KUAL00201
[root@master ~]# kubectl run kua100201 --image=redis --replicas=7 --labels=app_env_stage=dev --dry-run -o yaml > /opt/KUAL00201/deploy_spec.yaml
[root@master ~]# kubectl apply -f /opt/KUAL00201/deploy_spec.yaml
[root@master ~]# kubectl delete -f /opt/KUAL00201/deploy_spec.yaml
```

第十一题 统计Service中的pod（3%）

Create a file /opt/KUCC00302/kucc00302.txt that lists all pods that implement Service foo in Namespce production
The format of the file should be one pod name per line

创建一个文件/opt/KUCC00302/kucc00302.txt，
其中列出在Namespce 为Production中实现Service 为foo的所有pod
文件的格式应该是每行一个pod名称

```
[root@master ~]# mkdir /opt/KUCC00302
[root@master ~]# touch /opt/KUCC00302/kucc00302.txt
```

[root@master ~]# kubectl get svc --show-labels -n production | grep foo > /opt/KUCC00302/kucc00302.txt #手动删除其他多余内容，只需要保留第一列的Pod名称即可

第十二题 secret（9%）

Create a kubetnetes Secret as follows:
Name: super-secret
Credential: alice or username:bob
Create a Pod named **pod-secrets-via-file** using the redis image which **mounts a secret named super-secret at /secrets**
Create a second Pod named **pod-secrets-via-env** using the redis image,**which exports credential/username as TOPSECRET/CREDENTIALS**

创建一个secret，使用以下：
名字：super-secret
Credential：alice or username：bob
创建一个pod名为pod-secrets-via-file 使用redis镜像，挂载名为super-secret的挂载路径/secrets
使用redis镜像创建第二个Pod名称Pod-secrets-via-env，使用credential/username 的方式，对应的变量为：TOPSECRET/CREDENTIALS

[root@master ~]# kubectl create secret generic super-secret --from-literal=credential=alice --from-literal=username=bob
[root@master ~]# kubectl get secret
[root@master ~]# vi super-secret.yaml #打开官网，复制两个模板env修改

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-secrets-via-file
  labels:
    name: pod-secrets-via-file
spec:
  volumes:
  - name: super-secret
    secret:
      secretName: super-secret
  containers:
  - name: pod-secrets-via-file
    image: redis
```

```
    volumeMounts:
    - name: super-secret
      readOnly: true
      mountPath: /secrets

apiVersion: v1
kind: Pod
metadata:
  name: pod-secrets-via-env
spec:
  containers:
  - name: pod-secrets-via-env
    image: redis
    env:
      - name: TOPSECRET
        valueFrom:
          secretKeyRef:
            name: super-secret
            key: credential
      - name: CREDENTIALS
        valueFrom:
          secretKeyRef:
            name: super-secret
            key: username
#  restartPolicy: Never
[root@master ~]# kubectl apply -f ./super-secret.yaml
[root@master ~]# kubectl describe pod-secrets-via-file
[root@master ~]# kubectl describe pod-secrets-via-env
```

第十三题 EmptyDir (4%)

按如下要求创建一个 Pod：
名称：non-persistent-redis
Container image: redis
Persistent volume name: cache-control
Mount Path: /data/redis
应在 staging namespace 中发布，且该 volume 必须不能是永久的。

```
[root@master ~]# kubectl get ns
[root@master ~]# kubectl create ns staging
[root@master ~]# vi non-persistent-redis.yaml   #打开官网连接直接复制
```

```
apiVersion: v1
kind: Pod
metadata:
  name: non-persistent-redis
  namespace: staging
spec:
  containers:
  - image: redis
    name: non-persistent-redis
    volumeMounts:
    - mountPath: /data/redis
      name: cache-control
  volumes:
  - name: cache-control
    emptyDir: {}
```

```
[root@master ~]# kubectl apply -f non-persistent-redis.yaml
[root@master ~]# kubectl describe pod -n staging non-persistent-redis
```

第十四题 Scale：扩缩容 (1%)

Scale the deployment webserver to 6 pods

将部署webserver扩展到6个pod

```
[root@master ~]# kubectl get deployment
[root@master ~]# kubectl scale deployment/webserver --replicas=6
[root@master ~]# kubectl get deployment
```

第十五题 统计node是ready状态（2%）

check to see how many nodes are ready （not including nodes tained NoSchedule) and write the number to /opt/nodenum

检查有多少nodes是ready状态，（不包含node的污点，没有调度的），写入数量到/opt/nodenum

```
[root@master ~]# kubectl get nodes | grep Ready   #注意有多少行
```

[root@master ~]# kubectl describe node | grep Taints #注意NoSchedule的有多少行
[root@master ~]# echo 行数 > /opt/nodenum #Ready总行-NoScheule的行=最终有效行

第十六题 统计pod的CPU（2%）

From the pod label name=cpu-utilizer, find **pods running high CPU** workloads and write the name of the pod consuming most CPU to the file /opt/cpu.txt

从标签为name=cpu-utilizer的所有pod里面，找出cpu使用最高的那个pod，并写入到/opt/cpu.txt（这个文件已经存在）

[root@master ~]# kubectl get pod -A --show-labels | grep cpu-utilizer
[root@master ~]# kubectl top pod -n spacename podname
[root@master ~]# echo podname >> /opt/cpu.txt #只需要保留pod的名称即可

第十七题 nslookup查找service和pod的DNS记录（7%）

Create a deployment as follows
Name: nginx-dns
Exposed via a service: nginx-dns
Ensure that the service & pod are accessible via their respective DNS records
The container(s) within any pod(s) running as a part of this deployment should use the nginx image
Next,use the utiliity nslookup to look up the DNS records of the service & pod and write the output to **/opt/service.dns** and /opt/pod.dns respectively
Ensure you use the busybox:1.28 image (or earliser) for any testing, an the latest release has an unpstream bug which impacts the use of nslookup

创建一个deployment 名为nginx-dns，发布一个服务名为: nginx-dns
确保service 和pod可以通过各自的DNS记录访问
作为deployment的一部分运行的任何pod中的容器都应该使用nginx镜像
接下来，使用实用程序nslookup查找service和pod的DNS记录，并将输出分别写入/opt/service.dns和/opt/pod.dns
确保您使用busybox:1.28 image(或更早版本)进行任何测试，最新版本有一个unpstream bug，影响nslookup的使用。

[root@master ~]# kubectl run nginx-dns --image=nginx

[root@master ~]# kubectl expose deployment nginx-dns --port=80
[root@master ~]# kubectl run busybox -it --image=busybox:1.28 sh
[root@master ~]# kubectl exec -ti busyboxname -- nslookup nginx-dns > /opt/service.dns
[root@master ~]# kubectl exec -ti busyboxname -- nslookup <Pod ip> > /opt/pod.dns

第十八题 etcd快照（7%）

Create a snapshot of **the etcd instance running at http://127.0.0.1:2379 saving the snapshot to the file path /data/backup/etcd-snapshot.db**
The etcd instance is running etcd version 3.2.18
The following TLS certificates/key are supplied for connnecting to the server with etcdctl
CA certificate：/opt/KUCM00302/ca.crt
Client certificate：/opt/KUCM00302/etcd-client.crt
Client key: /opt/KUCM00302/etcd-client.key

[root@master ~]# etcdctl --endpoints=http://127.0.0.1:2379 --ca-file=/opt/KUCM00302/ca.crt --certfile=/opt/KUCM00302/etcd-client.crt --key=/opt/KUCM00302/etcd-client.key snapshot save /data/backup/etcd-snapshot.db

如有需要指定将输出结果保存到某一个文件
[root@master ~]# etcdctl snapshot status /etcd-snapshot.db > XXX.txt

第十九题 使用 kubectl drain 从集群中移除节点（4%）

Set configuration context $ kubectl config use-context ek8s
Set the node labelled with name=ek8s-node-1 as **unavailable** and **reschedule all the pods running on it**

把标签为name=ek8s-node-1的node设置为unavailable和重新安排所有运行在上面的pods

[root@master ~]# kubectl get nodes --show-labels -A | grep name=ek8s-node-1
[root@master ~]# kubectl drain nodename

注意：若要恢复节点，可以执行如下
[root@master ~]# kubectl uncordon

如果直接drain会出错，需要添加--ignore-daemonsets --delete-local-data参数

[root@master ~]#kubectl drain node node1 --ignore-daemonsets --delete-local-data

第二十题 修复notready状态 的节点（4%）

A Kubernetes worker node,labelled with name=wk8s-node-0 is in state NotReady.
Investigate why this is the case,and perform any appropriate steps to bring the node to a Ready state,ensuring that any chanages are made permanent.
Hints:
You can ssh to the failed node using $ ssh wk8s-node-0
You can assume elevated privileges on the node with the following command $ sudo -i

wk8s集群里面有一个标签为wk8s-node-0的节点是notready状态，找到原因，恢复Ready状态，所做的改变要是持久的

[root@master ~]# kubectl get node #查看哪个node为notready状态，ssh连接上去
[root@master ~]# sudo -i
[root@master ~]# systemctl status kubelet
[root@master ~]# systemctl start kubelet
[root@master ~]# systemctl enable kubelet

第二十一题 Create static Pods（4%）

configure **the kubelet systemed managed service, on the node labelled with name=wk8s-node-1,to launch a pod containing a single container of image nginx named myservice** automatically.
Any spec file requried should be placed in the **/etc/kubernetes/manifest**s directory on the node
Hints:
You can ssh to the failed node using $ **ssh wk8s-node-0**
You can assume elevated privileges on the node with the following command $ **sudo -i**

配置一个kubelet 系统管理的服务，在标签为name1=wk8s-node-1的节点上配置，要包含一个POD名为myservice的镜像nginx 容器。
所需的任何特定文件应放在/etc/kubernetes/manifests 的节点文件夹内

[root@node1 ~]# ssh wk8s-node-0
[root@node1 ~]# sudo -i

[root@node1 ~]# kubectl run myservice --image=nginx --generator=run-pod/v1 --dry-run -o yaml > /etc/kubernetes/manifests/wk8s-node-0.yaml
[root@node1 ~]# vi /var/lib/kubelet/config.yaml #查看staticPodPath: /etc/kubernetes/manifests
[root@node1 ~]# systemctl daemon-reload
[root@node1 ~]# systemctl restart kubelet
[root@node1 ~]# systemctl enable kubelet

@@@@第二十二题 给出一个集群，将节点node1添加到集群中（8%）

第二十三题 集群故障排查（kubelet配置的静态Pod路径） （4%）

Given a partially-funnctioning Kubernetes cluster,identify symptoms of failure on the cluster.
Determine the node,the failing service and take actions to bring up the failed service and restore the health of the cluster.
Ensure that any changes are made permanently.
The worker node in this cluster is labelled with name=bk8s-node-0

Hints:
You can ssh to the relevant nodes using $ **ssh $(NODE) where $(NODE) is one of bk8s-master-0 or bk8s-node-0**
You can assume elevated privileges on anynode in the cluster with the following command $ **sudo -i**

给定一个部分功能正常的Kubernetes集群，识别集群上的故障症状。
确定节点、failling服务器并采取启动失败的服务并恢复集群的健康状态，确保永久地进行任何更改。
这个集群中的工作节点被标记为name=bk8 -node-0

情况一：
[root@master ~]# kubectl get node,cs #查看是否是群集组件API问题，一般报6443.....
[root@master ~]# cd /etc/kubernetes/manifests/ #确定群集是否用kubeadmin方式部署的，存在文件
[root@master ~]# vi /var/lib/kubelet/config.yaml #确定staticPodPath路径是否正确
[root@master ~]# systemctl restart kubelet
[root@master ~]# kubectl get node,cs

情况二：

[root@master ~]# kubectl get node,cs

[root@master ~]# ssh master

[root@master ~]# systemctl restart kube-manager-controller.service

第二十四题 持久卷（3%）

Create a persistent volume with name app-config of capacity 1Gi and access mode ReadWriteOnce.

The type of volume is hostPath and its location is /srv/app-config

创建一个持久卷，名称为app-config，容量为1Gi，访问模式为ReadWriteOnce

卷的类型是hostPath，它的位置是/srv/app-config

[root@node1 ~]# vi persistent.yaml

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: app-config
spec:
  capacity:
    storage: 1Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: /srv/app-config
```

[root@node1 ~]# kubectl apply -f persistent.yaml